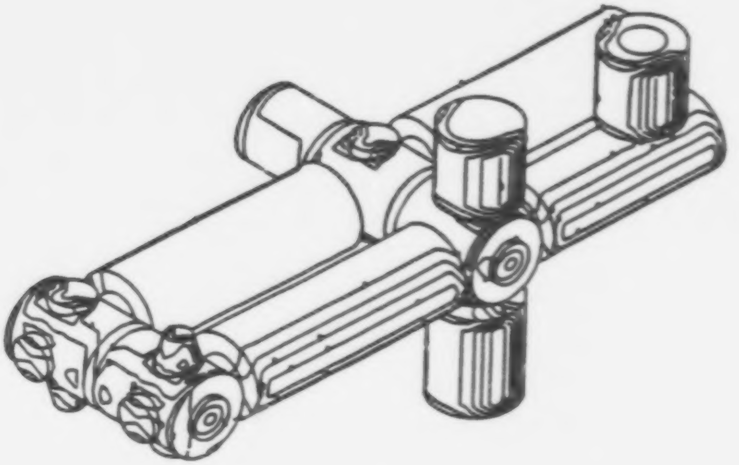
D180-30550-4

Users Guide
For Design Analysis Code BUMPER

SPACE STATION INTEGRATED WALL DESIGN AND PENETRATION DAMAGE CONTROL

by

A. R. Coronado, M. N. Gibbins, M. A. Wright, and P. H. Stern



Prepared for

National Aeronautics and Space Administration

July 1987

Contract NAS8-36426

Technical Management
NASA George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama
Structures and Dynamics Laboratory
Sherman L. Avans

Boeing Aerospace Company
Seattle, Washington

(NASA-CR-179169) SPACE STATION INTEGRATED N88-10070
WALL DESIGN AND PENETRATION DAMAGE CONTROL
(Boeing Aerospace Co.) 312 p Avail: NTIS
HC A14/MF A01 CSCL 22B Unclas
G3/18 0084733

D180-30550-4

Users Guide
For Design Analysis Code BUMPER

SPACE STATION INTEGRATED WALL DESIGN AND PENETRATION DAMAGE CONTROL

by

A. R. Coronado, M. N. Gibbins, M. A. Wright, and P. H. Stern

Prepared for

National Aeronautics and Space Administration

July 1987

Contract NAS8-36426

Technical Management
NASA George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama
Structures and Dynamics Laboratory
Sherman L. Avans

Boeing Aerospace Company
Seattle, Washington

CONTENTS

Page

1.0 INTRODUCTION 1

1.1 DOCUMENT ORGANIZATION/SECTION DESCRIPTION 1

2.0 BUILDING A MODEL 3

2.1 INPUT FILE FORMAT 3

2.2 MODELING CONVENTIONS 3

3.0 ANALYZING A MODEL 9

3.1 REFERENCE CONFIGURATION 9

3.2 GEOMETRY MODULE 9

3.3 RESPONSE MODULE 9

3.4 BUMPER MODULE 12

3.5 CONTOUR MODULE 12

4.0 PERFORMING SENSITIVITY STUDIES 13

4.1 MANUALLY CALCULATED FROM PREVIOUS RESULTS 13

4.2 STUDIES REQUIRING RERUNNING SOLUTION SEQUENCE 14

5.0 GENERATING CONTOUR PLOTS 17

5.1 THREAT CONTOUR PLOTS 17

5.2 DESIGN CONTOUR PLOTS 17

6.0 EXAMPLES 20

6.1 GEOMETRY 21

6.2 RESPONSE 21

6.3 BUMPER 29

6.4 CONTOUR 34

6.5 RPLOT 38

7.0 APPLICATION OF THEORY 41

Page

7.1 HISTORY 41

7.2 DETERMINING THE PROBABILITY OF NO PENETRATION 41

7.3 GEOMETRY VERSION 2.51 41

7.3.1 Write Header 43

7.3.2 Calculate the Threat Data 43

7.3.3 Read in the Model File 46

7.3.4 Calculate the Normal Vectors 46

7.3.5 Calculate the Centroid Locations 46

7.3.6 Calculate the Surface Areas 46

7.3.7 Calculate the Radius of a Sphere Enclosing the Elements 46

7.3.8 Eliminate Back Side Elements 46

7.3.9 Transform the Nodal Coordinates and Centroid Locations 46

7.3.10 Sort on the Transformed X-Dimension 46

7.3.11 Sort on the Transformed Y-Dimension 48

7.3.12 Eliminate the Shadowed Elements 48

7.3.12 Output List of Exposed Elements and Associated Impact Angles 48

7.4 RESPONSE VERSION 2.0 48

7.4.1 Set Increments 51

7.4.2 Read in User Inputs 51

7.4.3 Set the Impact Angle 51

7.4.4 Set the Impact Velocity 51

7.4.5 Single Plate Penetration Function 51

7.4.6 Double Plate Penetration Function 51

7.4.7 PEN4 54

Page

7.4.8 Write Output 54

7.5 BUMPER VERSION 4.0 54

7.5.1 Read User Inputs 60

7.5.2 Read in the Geometry Data Base 60

7.5.3 Read in the Response Data Base 60

7.5.4 Set the Threat Data 60

7.5.5 Set the Impact Conditions for Element Nel 60

7.5.6 Calculate the Critical Diameter 61

7.5.7 Calculate the Flux of the Critical Diameter 61

7.5.8 HISTORY(NEL) = HISTORY(NEL) + $F(D_c(v_i,\beta_i))$ * $COS(\beta_i)$ * $\rho(\alpha_i,\theta_i)$ 61

7.5.9 ARHIS(NEL) = ARHIS(NEL) + $COS(\beta_i)$ * $\rho(\alpha_i,\theta_i)$ 61

7.5.10 HISTORY(NEL) = HISTORY(NEL) * Area(NEL), ARHIS(NEL) = ARHIS(NEL) * Area(NEL) 61

7.5.11 Sum up the History and Arhis Arrays by Ranges 61

7.5.12 Calculate PNP for Each Range 61

7.5.13 Write PNP and Effective Area Out 62

7.5.14 SUPERTAB Output File 62

7.6 CONTOUR VERSION 2.0 62

7.6.1 Read in User inputs 62

7.6.2 Set Vessel Wall Thickness 62

7.6.3 Set Shield Thickness 62

7.6.4 Ratio OK 62

7.6.5 Build Response Array 64

7.6.6 Inside Range 64

7.6.7 Sum History by Specific Range and PID 64

Page

7.6.8 Calculate the PNP for the Specific Range and PID 64

7.6.9 Write Out Ts, S, Tb, PNP 64

REFERENCES 65

APPENDIX A - SUPERTAB Universal File Format 67

APPENDIX B - Source Code for GEOMETRY Module B-1

APPENDIX C - Source Code for RESPONSE Module C-1

APPENDIX D - Source Code for BUMPER Module D-1

APPENDIX E - Source Code for CONTOUR Module E-1

FIGURES

		Page
1.0-1	Analysis Flow	2
2.2-1	Modeling Conventions	4
2.2-2	Dynamic Receiver Model	6
3.1-1	Space Station Geometry Model, 5000 Elements	10
3.3-1	Correlations of Predictions Versus Test Data	11
4.1-1	Results of Orbital Time Sensitivity Study	15
4.2-1	Results of Orientation Sensitivity Study	16
5.1-1	Threat Contour Plot - Debris Environment	18
5.1-2	SUPERTAB Postprocessing Input for Threat Contour Plot	19
5.2-1	Design Contour Plot - Debris Environment	20
6.1-1	Example of GEOMETRY Run - Debris	22
6.1-2	Debris Threat Generation	23
6.1-3	Example of GEOMETRY Run - Meteoroids	24
6.1-4	Meteoroid Threat Generation	25
6.2-1	Example of RESPONSE Run	26
6.2-2	RESPONSE Summary File	28
6.3-1	Example of BUMPER Run	30
6.3-2	BUMPER Summary File	32
6.3-3	Example of Orbital Altitude Calculation	33
6.4-1	Example of CONTOUR Run	35
6.4-2	Sample CONTOUR Summary File	37
6.5-1	Example of RPLOT Run	39
6.5-2	Sample RPLOT Output File	40

		Page
7.3-1	GEOMETRY Flow Chart (Version 2.0)	44
7.3-2	Transformation of Velocity Distribution to Approach Angle Distribution	45
7.3-3	Transformation of Nodes and Sorting of Elements	47
7.3-4	Geometry Data Base File Structure	49
7.4-1	RESPONSE Flow Chart (Main Program.)	50
7.4-2	Example of MAT.PRP File (Material Property File.)	52
7.4-3	Comparison of Penetration Resistance to Meteoroids of Single Wall and Double Wall Structures	53
7.4-4	Penetration Function With Burch Equations	55
7.4-5	Regression Penetration Function	56
7.4-6	Penetration Function With PEN4	57
7.4-7	Response Data Base File Structure	58
7.5-1	BUMPER Flow Chart	59
7.6-1	CONTOUR Flow Chart	63

GLOSSARY

BAC	Boeing Aerospace Company
BUMPER	The family of computer codes developed under this contract or the specific module for calculating the probability of no penetration.
Critical Diameter	The diameter of a particle that just penetrates a configuration for a given impact velocity and angle.
EID	element identification number
FEM	finite element model
Flux	The number of particles passing through unit area per unit time.
IELM	Parameter controlling maximum number of elements in common block.
MLI	multilayer insulation
MSFC	Marshall Space Flight Center
NASA	National Aeronautics and Space Administration
NASTRAN	NASA structural analysis finite element code
NASUP	NASTRAN to universal file format translator
NEL	Nth element
PC	personal computer
PID	element's property identification number
PNP	probability of no penetration
RPLOT	translates RESPONSE binary output file to a formatted file suitable for plotting.
SUPERTAB	Graphic preprocessor and postprocessor for finite element analysis by Structural Dynamics Research Corporation of San Diego, California.
SSCE	Space Station critical element

Universal File Format	Machine-independent format for transmitting data to and from SUPERTAB (see app. A)
[illegible]	Mainframe computer build by Digital Equipment Corporation.
VMS	Virtual Memory System, operating system for VAX computers.

1.0 INTRODUCTION

The analysis code BUMPER executes a numerical solution to the problem of calculating the probability of no penetration (PNP) of a spacecraft subject to man-made orbital debris or meteoroids impact. The term "BUMPER" as used in this document, refers to both the overall family of codes as well as the specific computer program BUMPER. This document provides step-by-step procedures and theory for performing such an analysis. The examples provided in the text reflect our approach to analyzing Space Station structure, but the principles can be applied to any structure that can be modeled with finite elements. Advantages of this approach include (1) impact shielding of one element by another (shadowing) is considered, (2) users can specify various shield configurations over the spacecraft exterior to reflect design requirements, and (3) the effects of changing spacecraft flight orientation and orbital altitude can be determined.

The codes were developed on a DEC VAX 11/780 computer that uses the VMS operating system. They are written in Fortran 77 with no VAX extensions.

To help illustrate the steps involved, we carry a single sample analysis through the users guide. The example is our Space Station reference configuration used throughout contract performance. The finite element model (FEM) of this configuration is relatively complex but demonstrates many BUMPER features.

A flow chart for a complete analysis is shown in figure 1.0-1. The path indicated by solid lines is required to perform a fundamental analysis resulting in an overall PNP for the spacecraft. The dashed lines show how ancillary sensitivity studies on design variables are performed.

1.1 DOCUMENT ORGANIZATION/SECTION DESCRIPTIONS

Section 2.0 describes computer tools and guidelines for constructing a FEM for the space structure under consideration.

Section 3.0 carries through the steps in a fundamental analysis of the model constructed in section 2.0.

Section 4.0 describes the methods used to analyze the sensitivity of PNP to variations in design.

Section 5.0 suggests ways for developing contour plots of the sensitivity study data.

Section 6.0 provides additional BUMPER analysis examples and includes FEMs, command inputs, and data outputs.

Section 7.0 describes the mathematical theory used as the basis for the code and illustrates the data flow within the analysis.

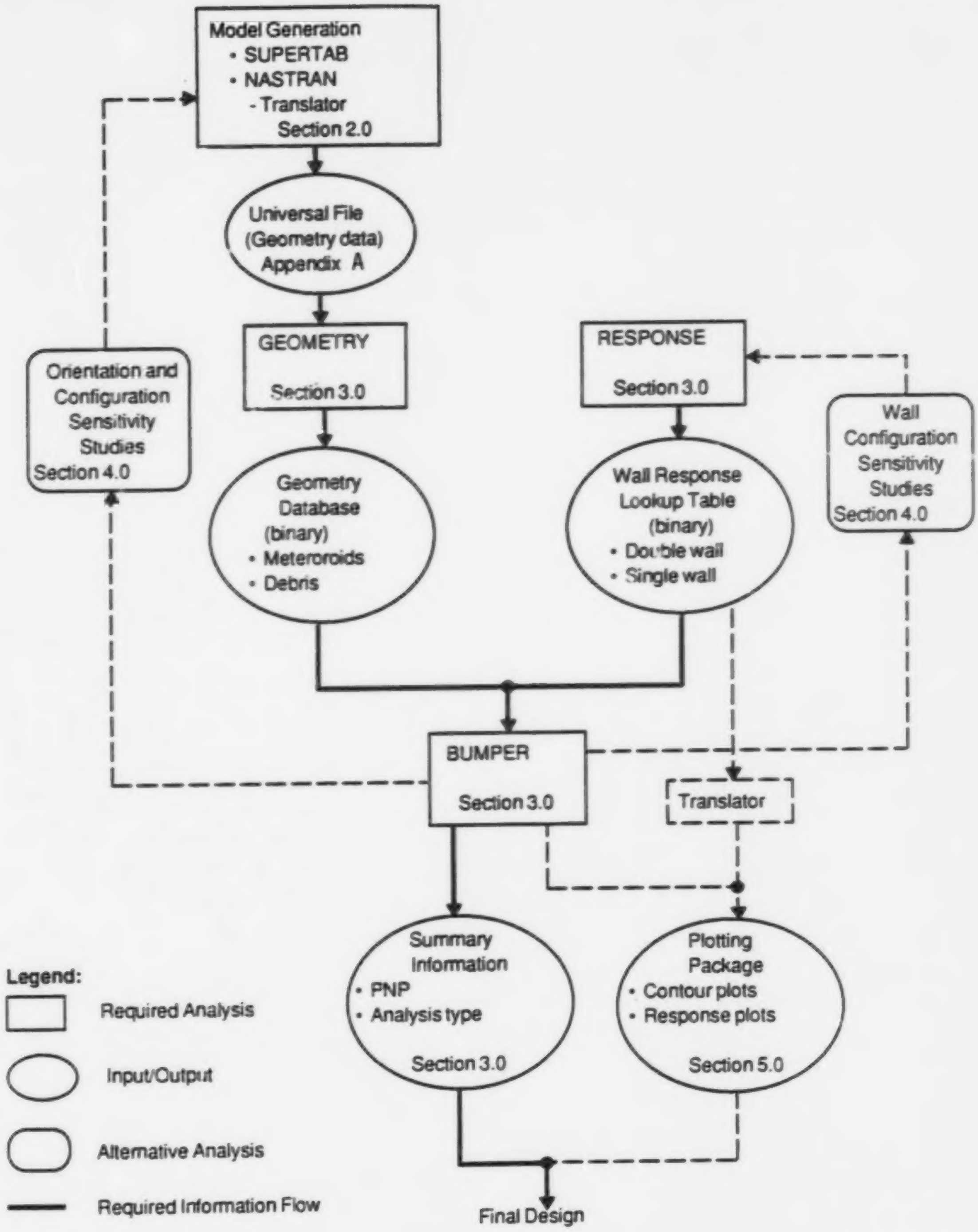

Model Generation
• SUPERTAB
• NASTRAN
- Translator
Section 2.0
Universal File
(Geometry data)
Appendix A
GEOMETRY
Section 3.0
RESPONSE
Section 3.0
Orientation and
Configuration
Sensitivity
Studies
Section 4.0
Wall
Configuration
Sensitivity
Studies
Section 4.0
Geometry
Database
(binary)
• Meteroroids
• Debris
Wall Response
Lookup Table
(binary)
• Double wall
• Single wall
BUMPER
Section 3.0
Translator
Summary
Information
• PNP
• Analysis type
Section 3.0
Plotting
Package
• Contour plots
• Response plots
Section 5.0
Final Design
Legend:
Required Analysis
Input/Output
Alternative Analysis
Required Information Flow
Optional Information Flow

Figure 1.0-1. Analysis Flow

870512-1

2.0 BUILDING A MODEL

The geometry and orientation of the spacecraft is defined mathmatically through the the use of a FEM. The conventions for building such a model are described below.

2.1 INPUT FILE FORMAT

The FEM model used to represent the spacecraft geometry and configuration may be built using conventional FEM generation techniques. Under this contract, the FEM preprocessor and postprocessor SUPERTAB was used for building and debugging the model and later for analysis results presentation. This was done for convenience; the model generation code or method used is not critical. Other FEM codes such as ANSYS or SPAR may be used to build the model but will require a translator to convert the node and element connectivity information into the proper format. After generation the completed model is written to an analysis input file in the SUPERTAB Universal File Format. This is the format the analysis code GEOMETRY uses. For convenience a NASTRAN to Universal File Format translator named NASUP is available from NASA/MSFC. A model built using another format will require a translator to convert it to the Universal File Format shown in appendix A of this document.

2.2 MODELING CONVENTIONS

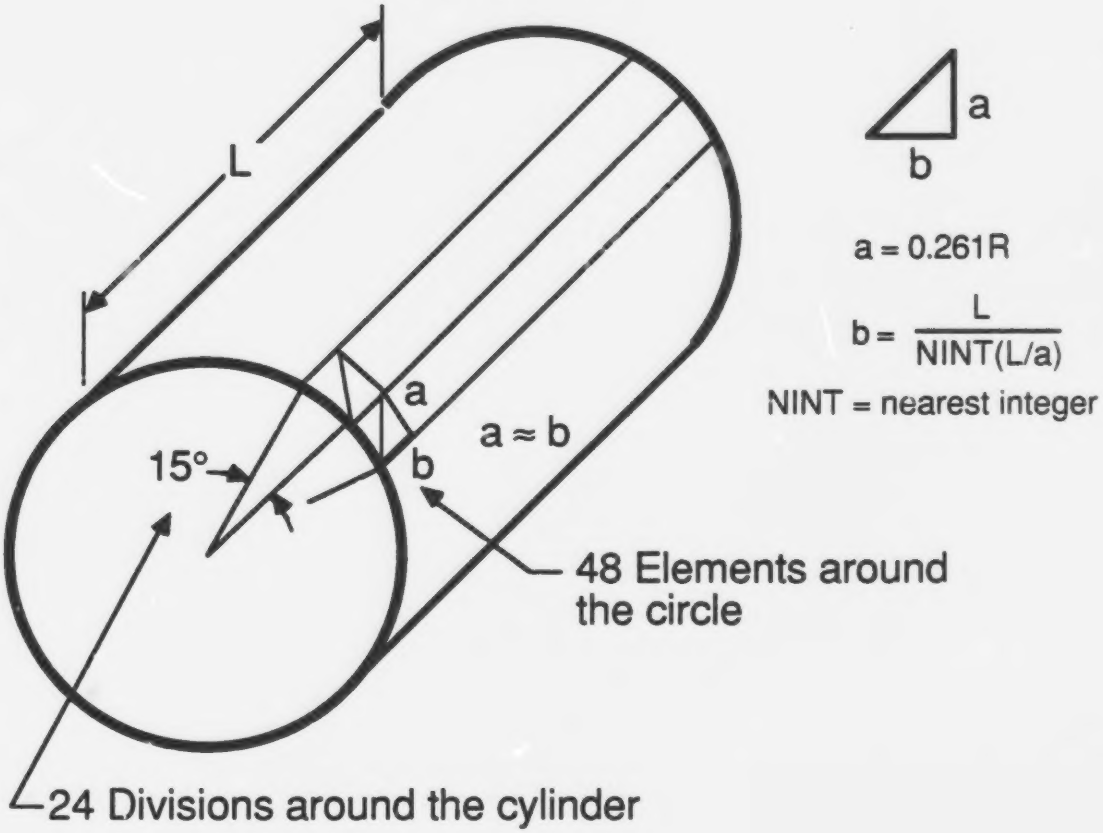
Nodes must all be defined in one global Cartesian coordinate system, as shown in figure 2.2-1. GEOMETRY does not check to see if more than one system is used. The positive X-axis must be parallel to the velocity vector, and the positive Z-axis must point toward the zenith (away from Earth). Failure to follow these conventions will invalidate the model and the analysis.

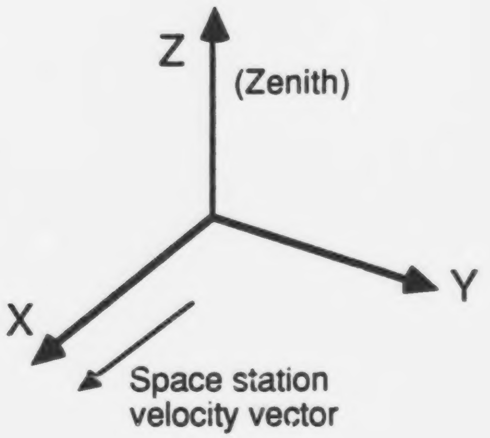
Only triangular elements may be use to build the FEM. The most reliable results are obtained using equilateral triangles, but right isosceles triangles may be more convenient when modeling cylindrical components. In any case, the element's aspect ratio should be as close to (1:1) as practical. Long, thin triangular elements may lead to erroneous results. This is because the hidden surface algorithm determines if an element's centroid is hidden by another element when deciding whether or not an element is exposed.

When modeling cylindrical sections, at least 24 divisions (48 elements) should be used around the circumference, as shown in figure 2.2-1. Axial divisions then should be chosen to optimize the aspect ratio (1:1). The user should be aware that sensitivity studies of element sizes were done for horizontal cylindrical sections only. Also, the user should ensure an adequate model by verifying that the calculated PNP does not change significantly with changes in mesh size.

All dimensions must be specified in meters. Using any other dimensional system will require code revisions.

The node numbering order defines, by the right-hand rule, the


L
a
b
a = 0.261R
b = L / NINT(L/a)
NINT = nearest integer
a
b
a ≈ b
15°
48 Elements around the circle
24 Divisions around the cylinder


Z
(Zenith)
Y
X
Space station velocity vector

Model Coordinate System

Figure 2.2-1. Modeling Conventions.

direction of the element's outward normal vector. The code uses the nodes in order 1, 2, 3. This vector is used to determine whether or not the element faces toward the threat, and it is extremely important to an accurate analysis. Ordinarily the outer surface of the FEM consists of elements all having outward normal vectors, but in special cases where impacts on internal surfaces (particles passing through openings) are important, additional elements with inward pointing normal vectors may be required. Figure 2.2-2 shows a Solar Dynamic Receiver model in which both outward and inward normal elements are used. Debris and meteoroids could strike the receiver's outer surface or pass through the orifice and strike the more sensitive inner surface.

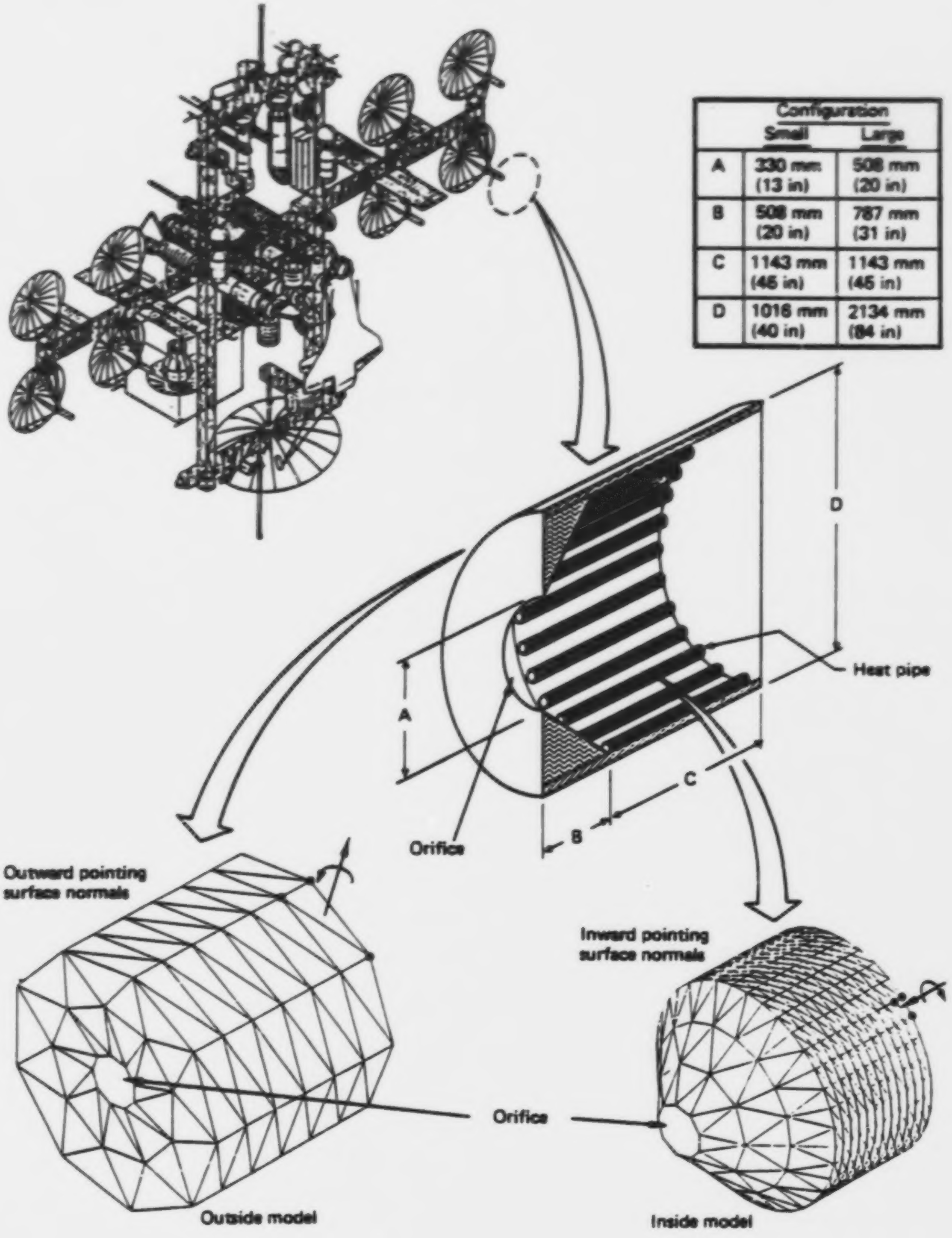
A maximum of 10 property identifications (PID), numbered 1 through 10 sequentially, may be defined. The code can be easily modified to allow for more PIDs. Each PID defines one unique wall, shield, spacing, and multilayer insulation (MLI) configuration. Therefore, each model may contain only 10 different wall configurations. These configurations may be either a conventional double plate (shield and pressure wall) arrangement, with or without 30 layers of MLI or a single thick wall. The code assumes all elements are parallel flat plates without curvature for penetration analysis.

The maximum number of elements allowed depends on the computer system. For a mainframe, the practical maximum is approximately 8000 elements; for an IBM-compatible personal computer (PC), the practical maximum is approximately 400 elements. The large data files required when analyzing a small model for meteoroid impacts may preclude using this code on a PC due to memory limitations.

The PNP will be calculated for specific ranges of element ids. Therefore several items should be kept in mind while building the model and determining the element numbering scheme. All elements representing a single Space Station critical element (SSCE) should be numbered within a single range of numbers and sequentially if possible, for example, number all the elements representing common module 1 in the range 1000-1999 and for node 3 in the range 5000-5999. This allows determination of PNP for each SSCE. Within this range of element IDs any combination of PIDs, up to a maximum of 10, may be used to accurately described differences in wall construction.

Many customary FEM modeling limitations do not apply here. Coincident, extra, and unattached nodes will not cause premature code failure. The model does not require constraint against translations or rotations, as it is used only to define spacecraft geometry. Structures that are in reality attached need only be modeled as adjacent to each other. An important limitation is that, if a structure normally shadows (hides) another's view of a threat, it must be modeled with sufficient accuracy and resolution to represent that shadowing. Any elements totally enclosed and therefore completely shielded from all threats by other elements will not affect the solution.

The limits on elements may be changed, if necessary, to allow


Heat pipe
D
A
C
B
Orifice
Outward pointing surface normals
Inward pointing surface normals
Orifice
Outside model
Inside model

	Configuration Small	 Large
A	330 mm (13 in)	508 mm (20 in)
B	508 mm (20 in)	787 mm (31 in)
C	1143 mm (45 in)	1143 mm (45 in)
D	1016 mm (40 in)	2134 mm (84 in)

Figure 2.2-2. Dynamic Receiver Model

analysis of larger models. This may be done by modifying the parameter IELM in the files COMMON1.BLK, and COMMON2.BLK and recompiling the GEOMETRY and BUMPER codes. A careful user should perform test cases to ensure numerical limitations or instabilities are not encountered. This can be done by rerunning the analysis for models previously run under the unmodified software to ensure no differences in the outputs occur.

To change the number of PIDs both the source code of RESPONSE and the COMMON2.BLK file must be modified. In the RESPONSE source code the dimension statement for RTABLE must be changed to the new values and the code recompiled. In the COMMON2.BLK file the dimensions of the RESPONSE array should match those of RTABLE. The BUMPER code then must be recompiled.

3.0 ANALYZING A MODEL

3.1 REFERENCE CONFIGURATION

To demonstrate the features of BUMPER and to illustrate design and analysis methodology, we selected the model of the reference configuration shown in figure 3.1-1; this model is described more fully in the Final Report (ref. 1). The reference configuration is representative of a typical Space Station configuration but does not represent any specific design.

Figure 1.0-1 shows the steps necessary to analyze the model described in section 2.0. Section 6.0 contains an example of the questions and menu options asked by each analysis module, what they represent, and typical responses. The required solution sequence is described in the following sections in the normal order of execution.

3.2 GEOMETRY MODULE

After the model has been built and verified, the geometry data must be written to a file using the SUPERTAB Universal File Format. GEOMETRY then reads in this information and creates the geometry data base for either meteoroid or debris analysis. Two geometry data bases are required for a complete analysis, one for debris and one for meteoroids. The data bases need only be recreated if the spacecraft geometry or orientation with respect to the velocity vector changes. This is the most computer-intensive portion of the analysis. If possible, it should be performed using a batch mode to minimize cost.

3.3 RESPONSE MODULE

After the geometry data bases have been produced, the wall penetration response data bases are created. Two data bases are required for a complete analysis, one for debris and one for meteoroids. RESPONSE creates these data bases for up to 10 unique wall configurations. These configurations are identified through the use of element property identifications (PID) in the FEM. RESPONSE does not have available any information about the model; therefore, the user must keep track of the number of PIDs and their relationship to the actual spacecraft. RESPONSE will prompt the user with various questions to determine the wall configurations for a particular analysis type. The results of the analysis are written to a user-defined file in binary format. The file contains the critical diameter for each wall configuration as a function of impact velocity and impact angle. A computer program named RPLOT is available from MSFC that converts the binary information to a form suitable for plotting. This is useful for illustrating the correlation between predicted wall response versus test data, as shown in figure 3.3-1. As a general rule, the user should examine the response plots for each wall configuration to ensure that the results are reasonable. RESPONSE is relatively inexpensive to use and therefore normally is run in an interactive mode. Examples of typical questions and inputs to RESPONSE and RPLOT are given in Section 6.0.

PRECEDING PAGE BLANK NOT FILMED

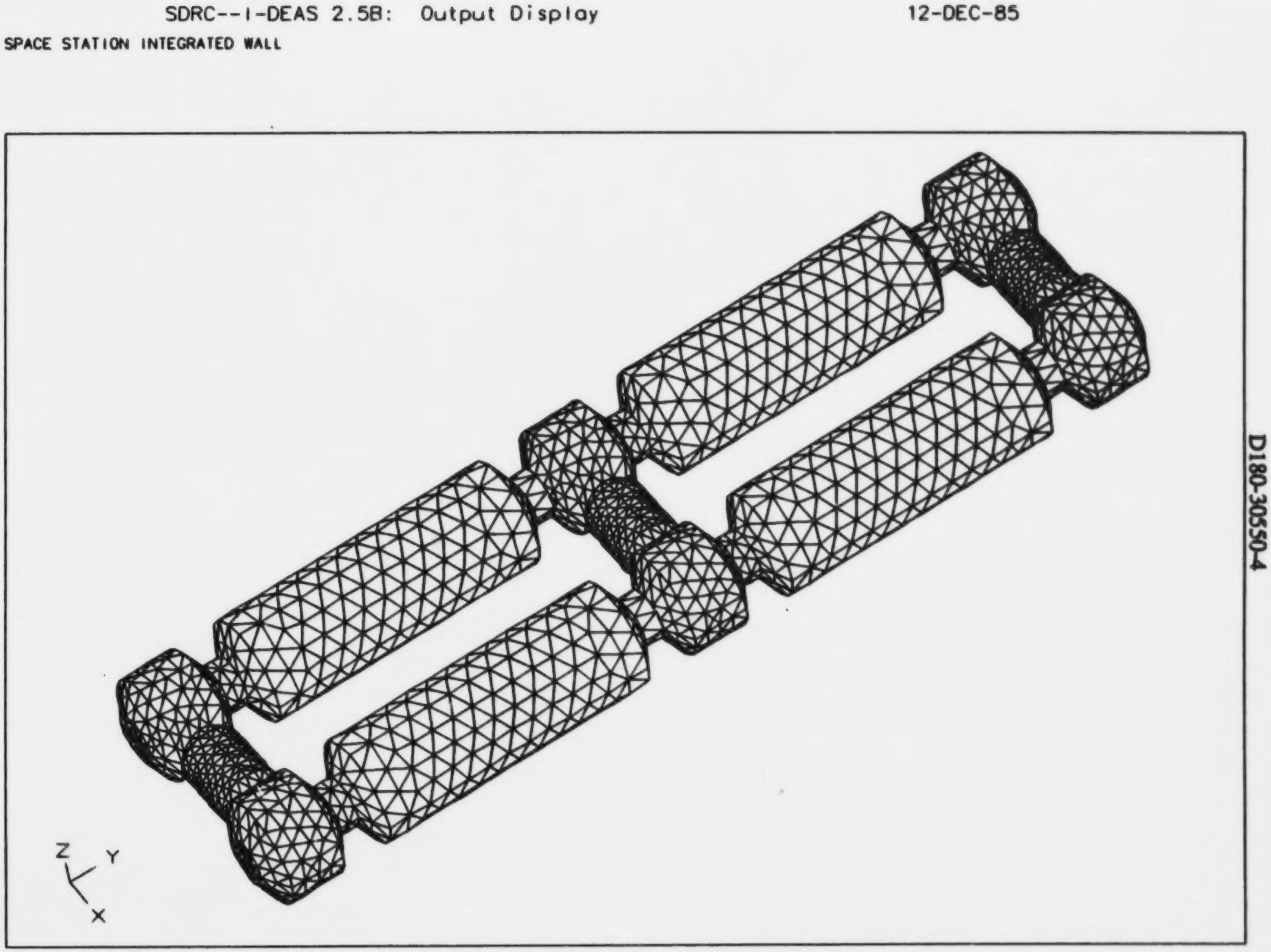

SDRC--I-DEAS 2.5B: Output Display
12-DEC-85
SPACE STATION INTEGRATED WALL
Z
Y
X

Figure 3.1-1. Space Station Geometry Model, 5000 Elements

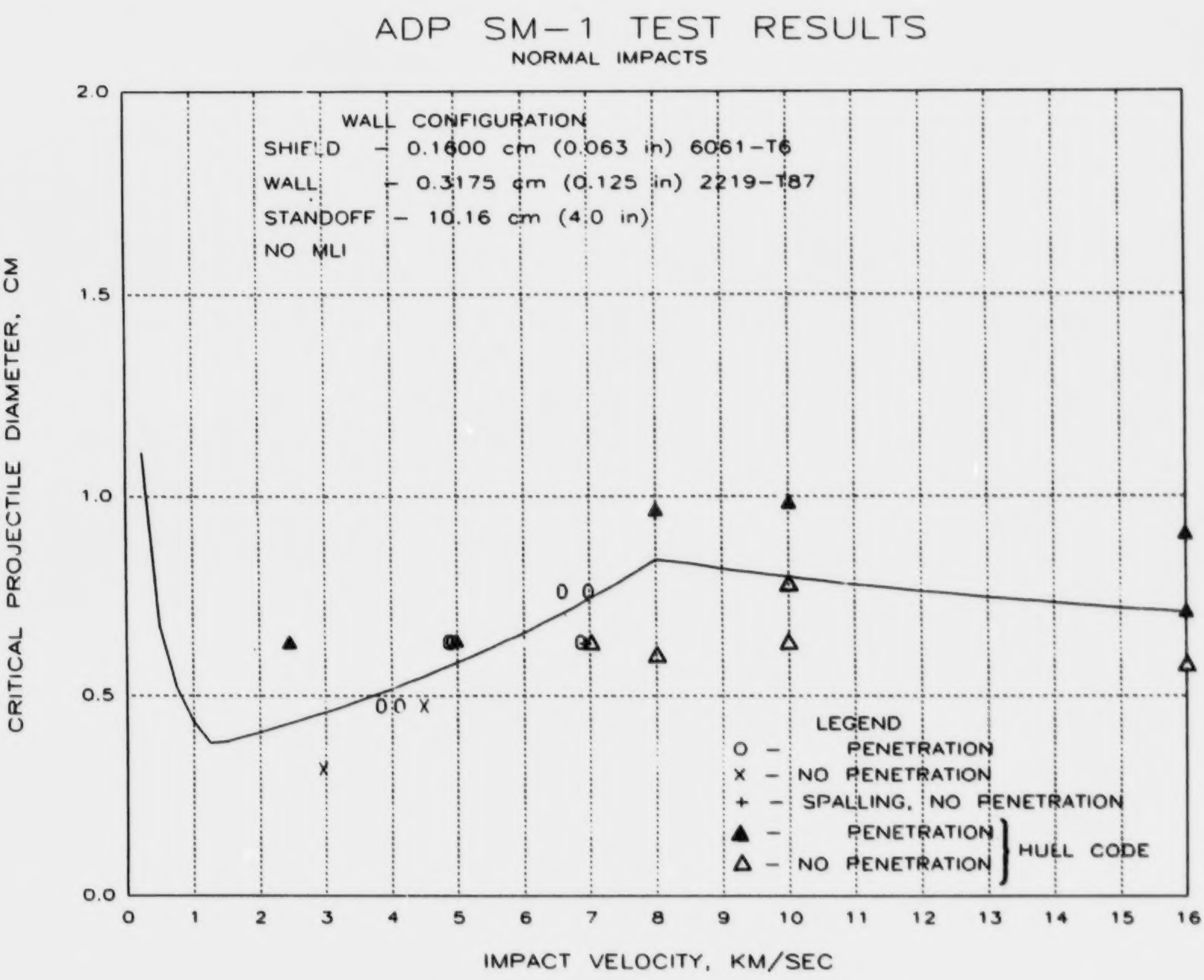


Figure 3.3-1: Correlations of Predictions Versus Test Data

3.4 BUMPER MODULE

Once the geometry and response data bases have been created, the probability of no penetration (PNP) is calculated using the BUMPER module. BUMPER will have to be run twice to perform a complete analysis, once for debris and once for meteoroids. The results from the two analyses are multiplied together to determine the overall PNP for both debris and meteoroids. The code prompts the user for information concerning the analysis type, summary filename, exposure time, operating altitude, geometry and response data base filenames, and the element ID ranges. The element ID ranges define the individual components (SSCE) for PNP calculation. The code also determines the PNP for the overall model. The results are then written both to the screen and to the user-defined summary file. Additionally BUMPER determines the effective area of the components and overall model. This information can be used in a simplified analysis to determine the number of particles of diameter D or greater per unit time affecting a specific component.

Additionally, BUMPER can produce the probability of penetration values per surface area for each element, which SUPERTAB can then plot as contours on the geometry model. These contours show the relative vulnerability of various Space Station external areas. This procedure is discussed in more detail in section 5.0

3.5 CONTOUR MODULE

BUMPER calculates the PNP for each component for a specific wall configuration. Sometimes, however, it is required to know how changes in wall configuration affect the PNP of an individual component. CONTOUR produces a data base that may be used with user-supplied software to produce design contour plots. These plots show the relationship between shield and vessel wall thickness and PNP for a fixed shield standoff.

The code is similar to BUMPER, with the exception being that the RESPONSE code has been incorporated as a subroutine. The code uses the same geometry data base as BUMPER. CONTOUR, however, is limited to the case of one PID and one element ID range. This topic is discussed further in section 5.0.

4.0 PERFORMING SENSITIVITY STUDIES

Two types of sensitivity studies are typically required, (1) those that can be calculated manually from previous results and (2) those that require rerunning part of the solution sequence. Both will be covered in this section.

4.1 MANUALLY CALCULATED FROM PREVIOUS RESULTS

Studies falling into this category measure the effects on PNP of simple variations in orbital time, surface area, or flux. To begin, a PNP value must be calculated using the solution sequence shown in figure 1.0-1 and described in section 3.0. The fundamental equation for calculating PNP is

$$\mathrm{PNP} = \mathrm{EXP}\,[\,-\lambda * T\,] \tag{1}$$

where λ is the average number of penetrating particles per unit time and T is the exposure time. λ is defined by equation 2

$$\lambda = \quad A_j * \quad \mathrm{COS}(B_i) * F(Dc(V_i,B_i)) * P(\theta_i,\alpha_i) * I_s \tag{2}$$

where A_j is the element's surface area, $F(D_c(V_i,B_i))$ is the flux of the critical projectile D_c with impact velocity V_i and impact angle B_i, $P(\theta_i,\alpha_i)$ is the probability of the threat described by polar angles Λ_i and α_i occurring and I_s is an indicator equal to 1 if the element is exposed and 0 if not. NT is the total number of threat angle cases and NELM is the number of elements in the range of interest (SSCE).

Scaler changes to the A, T, and F terms allow the PNP to be recalculated manually. Scaler changes to A imply that all the elements area change by a constant amount. Scaler changes to F imply that the flux of all particles changes by a constant amount. Scaler changes to T imply a simple change in the exposure time. Any changes that would affect the geometry or orientation of the model require that the GEOMETRY module be run again.

The preceding variables can be adjusted to calculate alternative PNPs. For example, if BUMPER calculated a PNP = 99% for a given Space Station configuration and orientation for 10 years in orbit, and it was necessary to calculate the effect of doubling the time in orbit, then the following technique should be used:

a. Divided PNP by 100 to convert it to a decimal fraction.
   PNP = 99%/100 = 0.99
b. Take the natural logarithm of the fractional form of PNP.
   Ln(0.99) = -0.0101 = $-\lambda T$
c. Divide $-\lambda T$ by 10 years to calculate the λ term.
   $-\lambda = -0.0101/10 = -0.00101$
d. Multiply the $-\lambda$ term by 20 years.
   $-\lambda T = -0.00101 * 20 = -0.0201$
e. Calculate a new PNP based on the new time in orbit.
   $\mathrm{PNP}' = 100\% * e^{-0.0201} = 98.01\%$

An example of this type of sensitivity study is shown in figure 4.1-1. This calculation is valid only if all other variables remain constant during the extended time in orbit. If the environment, surface area, or configuration is expected to change during that time, the technique in section 4.2 must be used. The previously described method can be used to study the effects of changes in either flux, time, or surface area with just a minimum of computing resources. (Note: doubling the value in step c produces an equivalent result by effectively doubling the time in orbit.)

4.2 STUDIES REQUIRING RERUNNING SOLUTION SEQUENCE

Studying changes in the Space Station's orientation, as shown in figure 4.2-1, requires a unique GEOMETRY analysis for each orientation. This requires a unique universal file with the nodal coordinates defined appropriately. SUPERTAB allows easy modification of the reference coordinate system, simplifying this task to a few keystrokes. The actual method used will depend on the model generation system selected by the user.

Sensitivity studies of changes in wall configuration require running GEOMETRY once to create either a debris or meteoroid data base and running RESPONSE once for each new wall configuration. BUMPER is then run for each unique response data base.

Sensitivity studies on the effects of changes to either the flux or the penetration function will require modification of either the RESPONSE or BUMPER modules, respectively. This will require careful study of section 7.0 and appendix C or D. If the changes in the flux are not constant, then the flux equation must be changed in the appropriate BUMPER subroutine. Changes in the penetration function must be handled in a similar manner, by modifying the appropriate equations in the various subroutines of RESPONSE.

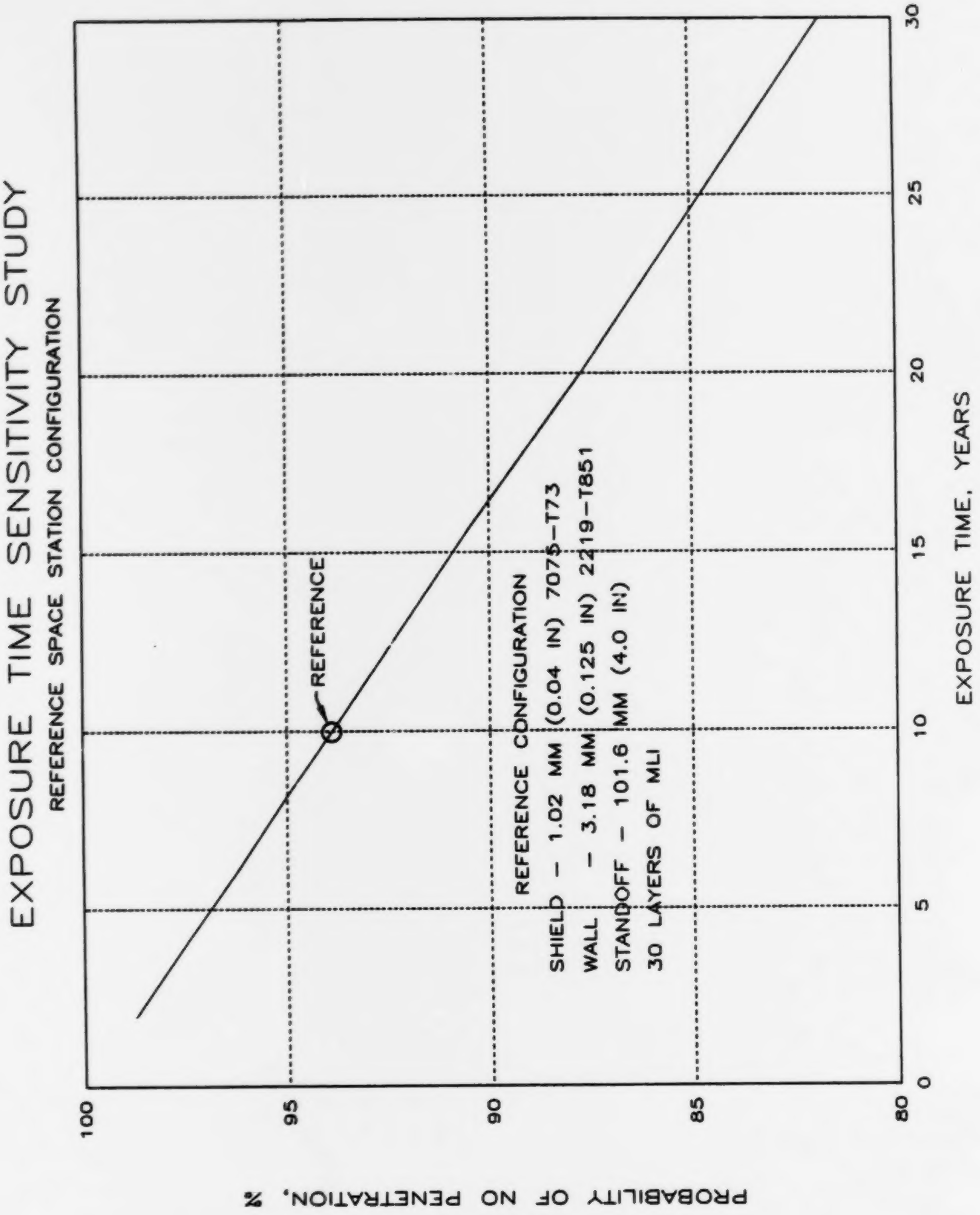


Figure 4.1-1. Results of Orbital Time Sensitivity Study

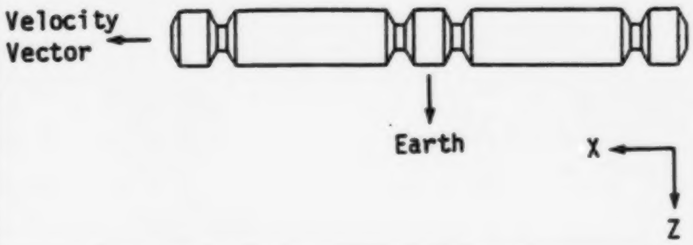
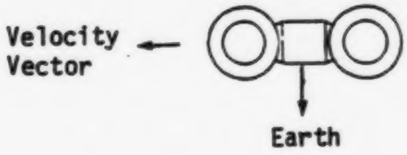
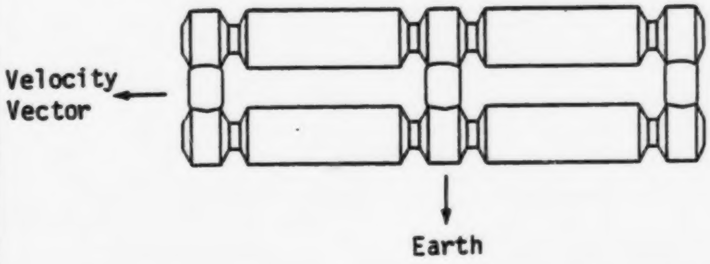
Orientations	Probability of No Penetration
Reference Configurations  Velocity Vector Earth X Z	93.4%
Rotation 90° Around Z Axis  Velocity Vector Earth	95.3%
Rotation 90° Around X Axis  Velocity Vector Earth	87.8%

Figure 4.2-1. Results of Orientation Sensitivity Study

5.0 GENERATING CONTOUR PLOTS

The data for two types of contour plots can be generated by BUMPER: (1) a threat contour plot and (2) a design contour plot. Nevertheless, the actual method of displaying the data generated by BUMPER depends on the computing and graphics resources available to user.

5.1 THREAT CONTOUR PLOTS

The threat contour plots are generated as a postprocessing feature of SUPERTAB. The last question BUMPER asks during execution is whether the user wants to write out a SUPERTAB input file. If the user answers "YES" BUMPER writes out a file containing the probability of penetration per surface area for each element in the model. An example of a threat contour plot is shown in figure 5.1-1. Figure 5.1-2 shows a sample SUPERTAB input file written by BUMPER for generating the plot in figure 5.1-1. An explanation of the format used is shown in appendix A.

Use of preprocessors other than SUPERTAB will require either the developing a translator to change the data format or modifying subroutine SUPER in BUMPER.

5.2 DESIGN CONTOUR PLOTS

Generating design contours is a more involved process. The range of input variables must be defined by determining a minimum, maximum, and increment for shield thickness (Ts), vessel wall (Tb) thickness, and spacing (S) between the two. Examples of inputs to generate this data are given in section 6.0. The program CONTOUR then loops through the parameters, calculating the PNP for each configuration and writing Ts, Tb, S, PNP to the summary file. Figure 5.2-1 shows a sample design contour plot, and figure 6.4-2 shows the associated CONTOUR output file used to generate such a plot.

The actual program used to produce the contour plot is a user-supplied item. During the contract a Boeing-developed isoterm graphics program was used but any contouring package should work. Care should be taken when selecting a package. Most packages use a linear interpolation to determine the contours. If this is the case the log of the PNP should be plotted versus shield and vessel wall thickness because it more closely varies linearly.

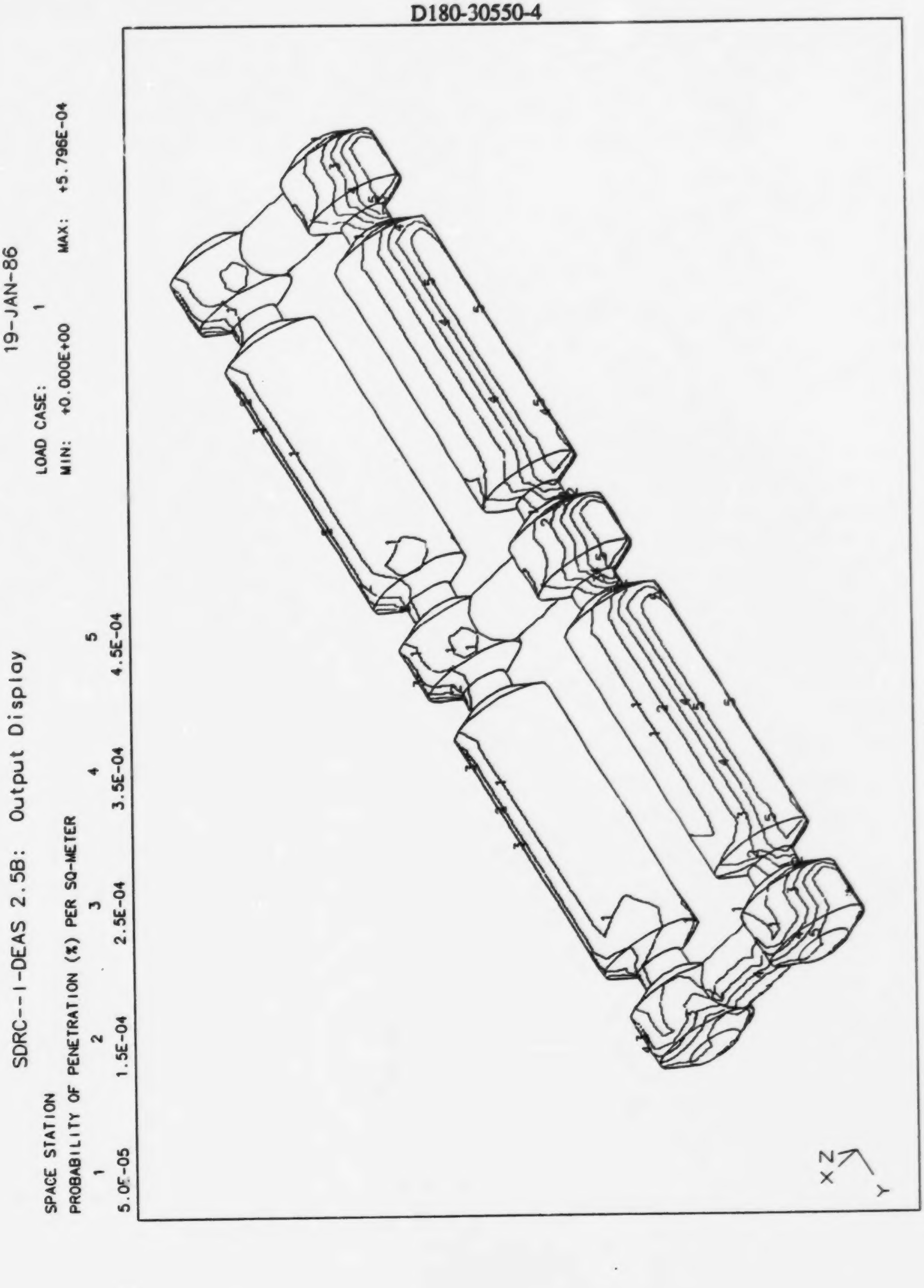


Figure 5.1-1. Threat Contour Plot – Debris Environment

```
   -1
   56
MAN-MADE ORBITAL DEBRIS ANALYSIS
PROBABILITY OF PENETRATION (%) PER SQ-METER
ONE
ONE
ONE
        1         0         1         1         2         1
        1         1         1
 0.00000E+00
    10000         1
 0.11814E-05
    10005         1
 0.30171E-04
    10020         1
 0.21931E-02
    10025         1
 0.21931E-02
    10030         1
 0.21931E-02
    10035         1
 0.21931E-02
    10040         1
 0.21931E-02
           .      .
           .      .
           .      .
           .      .
    10045         1
    85780         1
 0.18242E-02
    85790         1
 0.18242E-02
   -1
```

Figure 5.1-2. SUPERTAB Postprocessing Input for Threat Contout Plot

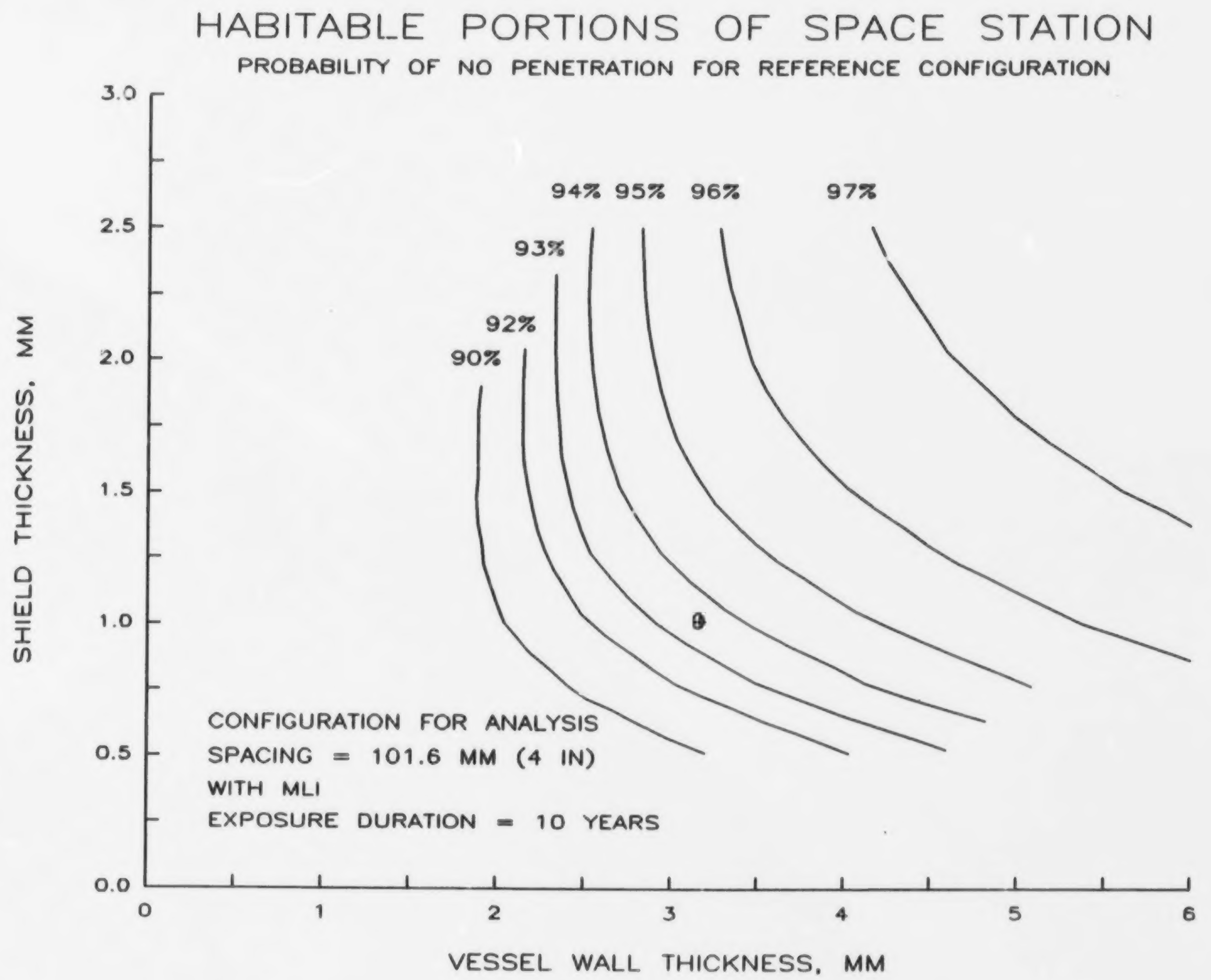


Figure 5.2-1. Design Contour Plot – Debris Environment

6.0 EXAMPLES

This section shows a sample of the execution of each module within BUMPER; a sample of the summary output file, and an explanation of each question as well as typical replies. Items in bold font are typed in by the user either to run a module or in response to questions.

6.1 GEOMETRY

Figure 6.1-1 shows the questions asked by GEOMETRY Version 2.51. The first question asks whether the geometry data base will be for debris or meteoroid analysis with the default being debris.

The second question asks for the name of the file containing the geometry data in a SUPERTAB-compatible Universal File Format. A carriage return defaults the file name to "STATION.UNI."

The third question asks for the data base file name. GEOMETRY will create this file and write out in a binary format the geometry data base. A carriage return defaults the data base filename to "STATION.GEM."

For debris analysis the fourth and final question asks for the number of uniform threats. The default is 45, which means that GEOMETRY will examine the model in 45 equal steps from +90.0 deg to -90.0 deg (every 4 deg) on either side of the X-axis or velocity vector, as shown in figure 6.1-2. GEOMETRY writes out a message as each threat case is completed and then indicates that it has successfully completed its function by writing out the name of the data base file.

If the desired analysis type was meteoroids, then the fourth question is about the number of uniform meteoroid threats. Figure 6.1-3 shows an example of a GEOMETRY run for meteoroid analysis. Figure 6.1-4 shows how a hemisphere is divided into approximately equal segments to generate a uniform series of meteoroid threats.

6.2 RESPONSE

Figure 6.2-1 shows the questions asked by RESPONSE Version 2.0 and gives typical responses. The first question asked, is if the threat to analyze is debris or meteoroids. The default is debris which means an aluminum projectile will be assumed. If meteoroids are chosen, an icy meteoroid projectile with a density of 0.5 g/c^3 will be assumed.

The second question asks for the wall property output file name. This summary file, shown in figure 6.2-2, records the analysis type and wall configuration used.

The third question asks for the response data base output file name. This is the binary data base contains the critical projectile diameter values as a function of PID, impact angle, and impact velocity. This file is then used by BUMPER to determine PNP.

```
RUN GEOMETRY

GEOMETRY VER 2.51

ANALYSIS TYPE ?
 1-DEBRIS <CR>
 2-METEOROIDS
ANSWER 1 OR 2 >1

SUPERTAB UNIVERSAL FILENAME (CR=STATION.UNI) >STATION.UNI

OUTPUT FILENAME (CR=STATION.GEM) >EXAMPLE.GEM

NUMBER OF UNIFORM DEBRIS THREATS  (CR=45) >45

THREAT CASE    1 COMPLETED
THREAT CASE    2 COMPLETED
THREAT CASE    3 COMPLETED
  .      .     .    .    .
  .      .     .    .    .
  .      .     .    .    .
THREAT CASE   42 COMPLETED
THREAT CASE   43 COMPLETED
THREAT CASE   44 COMPLETED
THREAT CASE   45 COMPLETED

OUTPUT LOCATED IN FILE EXAMPLE.GEM


$
```

Figure 6.1-1. Example of GEOMETRY Run — Debris

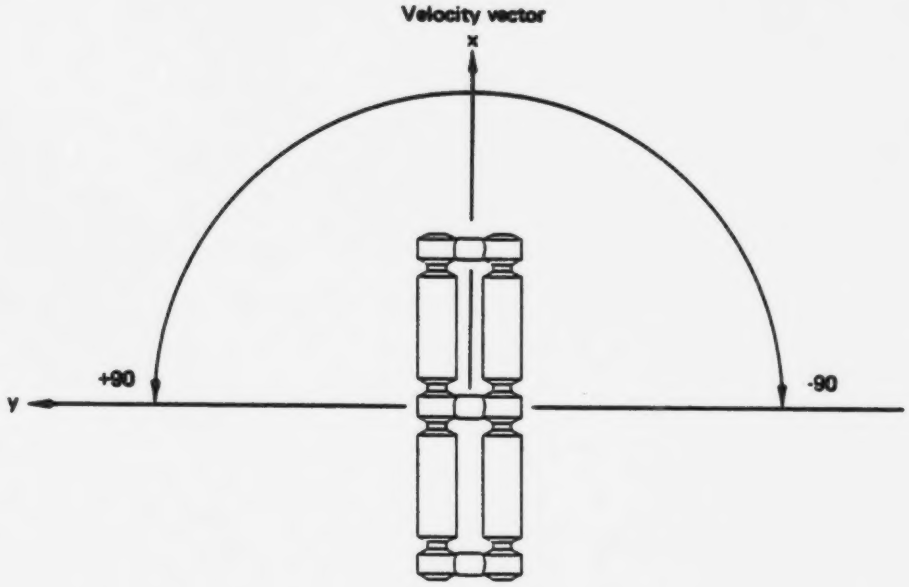

Velocity vector
x
+90
-90
y

Figure 6.1-2. Debris Threat Generation

```
RUN GEOMETRY

GEOMETRY VER 2.51

ANALYSIS TYPE ?
 1-DEBRIS <CR>
 2-METEOROIDS
ANSWER 1 OR 2 >2

SUPERTAB UNIVERSAL FILENAME (CR=STATION.UNI) >TUBES.UNI

OUTPUT FILENAME (CR=STATION.GEM) >EXAMPLE-2.GEM

NUMBER OF UNIFORM METEOROID THREATS ?
    1 - 84
    2 - 146<CR>
    3 - 232
    4 - 329
ANSWER (1-4) >2

THREAT CASE    1 COMPLETED
THREAT CASE    2 COMPLETED
THREAT CASE    3 COMPLETED
   .     .     .     .
   .     .     .     .
   .     .     .     .
   .     .     .     .
THREAT CASE  143 COMPLETED
THREAT CASE  144 COMPLETED
THREAT CASE  145 COMPLETED
THREAT CASE  146 COMPLETED

OUTPUT LOCATED IN FILE EXAMPLE-2.GEM


$
```

Figure 6.1-3. Example of GEOMETRY Run — Meteoroids

- Element coordinates selected to provide elements of equal area.
- Threat angles θ_1 and ϕ_1 are measured to centroid of curved surface element.
- All combinations of θ_1 and ϕ_1 are equally likely.

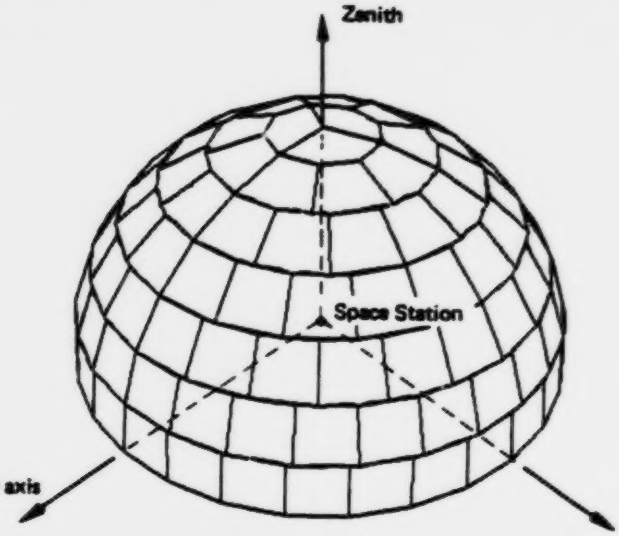

Zenith
Space Station
Reference axis

Figure 6.1-4. Meteroid Threat Generation

```
RUN RESPONSE

********************

 RESPONSE VER 2.0

Last Update 5/25/87

*******************

ANALYSIS TYPE ?
 1-DEBRIS <CR>
 2-METEOROIDS
ANSWER 1 OR 2 : 1

OUTPUT FILENAME FOR WALL PROPERTIES  <CR>=RESPONSE.SUM : RESPONSE.SUM

OUTPUT FILENAME FOR RESPONSE TABLES <CR>=STATION.RSP : EXAMPLE.RSP

INPUT IN METRIC OR ENGLISH UNITS  <CR>=METRIC : METRIC

PROPERTY ID NUMBER =  1

CONFIGURATION TYPE
    1- SINGLE PLATE
    2- DOUBLE PLATE <CR>
ANSWER (1 or 2) : 2

PENETRATION FUNCTION
    1-ORGINAL <CR>
    2-PEN4
    3-REGRESSION
ANSWER (1-3) : 3

SHIELD MATERIAL
   1- 2024-T4
   2- 2219-T87
   3- 6061-T6
   4- 7075-T6
SELECT MATERIAL NUMBER <CR>=1 : 3

SHIELD THICKNESS (CM) = : 0.160
```

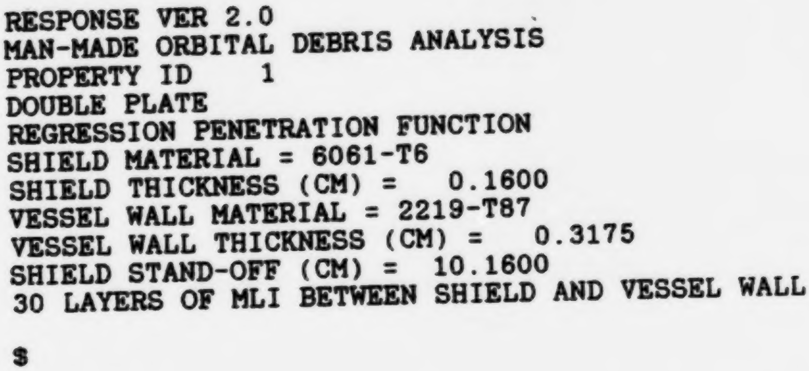
(continued on next page)

Figure 6.2-1. Example of RESPONSE Run

(continued from previous page)

```
VESSEL WALL MATERIAL
   1- 2024-T4
   2- 2219-T87
   3- 6061-T6
   4- 7075-T6
SELECT MATERIAL NUMBER <CR>=1 : 2

VESSEL WALL THICKNESS (CM) = : 0.3175

SHIELD STAND-OFF (CM) = : 10.160

INCLUDE 30 LAYERS OF MLI BETWEEN PLATES <CR>=YES : YES

DO YOU WISH TO RUN ANOTHER CASE ? <CR>=YES : NO
```

Figure 6.2-1. Example of RESPONSE Run (Continued)



```
RESPONSE VER 2.0
MAN-MADE ORBITAL DEBRIS ANALYSIS
PROPERTY ID     1
DOUBLE PLATE
REGRESSION PENETRATION FUNCTION
SHIELD MATERIAL = 6061-T6
SHIELD THICKNESS (CM) =    0.1600
VESSEL WALL MATERIAL = 2219-T87
VESSEL WALL THICKNESS (CM) =    0.3175
SHIELD STAND-OFF (CM) =  10.1600
30 LAYERS OF MLI BETWEEN SHIELD AND VESSEL WALL

$
```

Figure 6.2-2. RESPONSE Summary File

The fourth question determines the length units to be used for input-output, either English (inches) or metric (centimeters).

(The code loops through the next series of questions generating data bases for each unique wall configuration.)

The fifth question determines the configuration type, either a single plate wall construction or a dual plate consisting of a shield (bumper) and pressure wall, the latter is the default.

The sixth question is for double plate analysis only and determines the penetration function to be used. These functions are described in more detail both in reference 1 and in section 7.4.6. The recommended penetration function for Space Station type applications is regression.

The next series of questions establish the material and plate thickness for the shield and vessel (pressure) wall, the standoff distance between the two plates (for dual plate configuration), and whether or not 30 layers of MLI are to be included between the plates. For single plate analysis only, the vessel wall material and thickness is read in.

The last question asks whether the user wants to run another case (configuration). If the answer is "NO," then the code writes the data base and summary output file; otherwise it increases the property identification number (PID) in increments of one and resumes asking questions about the wall configuration. The default answers on subsequent wall configuration questions is the same as the answer on the previous configuration. The user must trace the number of PIDs and where they appear in the model.

6.3 BUMPER

Figure 6.3-1 shows the questions asked by BUMPER Version 4.0 and gives typical responses.

The first question asks for the name of the summary file, an example of which is shown in figure 6.3-2. As with RESPONSE, this summary output file can be used by the user as a record of the filenames, analysis, and results.

The second question establishes the analysis type, either debris or meteoroids. This determines which flux equation is used within BUMPER.

The third question asks for the exposure time (orbit time) in years. The default is 10 years.

The fourth question asks for the operating altitude of the spacecraft in kilometers. When the altitude varies with time (which is the case with Space Station), the recommended altitude is the antilogarithm of the sum of the logarithms (base 10) of the average yearly altitude. An example of the orbital altitude calculation is shown in figure 6.3-3. If this information is not available, then the average or nominal altitude should be used.

```
RUN BUMPER

*********************

   BUMPER VER 4.0

Last Update 5/25/87

*********************

SUMMARY OUTPUT FILENAME (CR=BUMPER.SUM)>

ANALYSIS TYPE ?
 1-DEBRIS <CR>
 2-METEOROIDS
ANSWER 1 OR 2 >1

SPACE STATION EXPOSURE TIME (YEARS) <CR=10.0> : 10.0

OPERATING ALTITUDE ( 400-500 km )  <CR=500> : 500.0

THE PROBABILITY OF NO PENETRATION WILL BE CALCULATED
FOR SPECIFIC RANGES OF ELEMENT IDS
INPUT THE STARTING AND ENDING ELEMENT ID FOR EACH RANGE

ENTER D <CR> OR <CR> WHEN DONE

RANGE    1
STARTING ELEMENT ID : 1

ENDING ELEMENT ID    : 10000

RANGE    2
STARTING ELEMENT ID :

GEOMETRY OUTPUT FILENAME ? <CR=STATION.GEM> >EXAMPLE.GEM

RESPONSE OUTPUT FILENAME ? <CR=STATION.RSP> >EXAMPLE.RSP
```

(continued on next page)

Figure 6.3-1. Example of BUMPER Run

```
     (continued from previous page)

THREAT CASE    1 COMPLETED
THREAT CASE    2 COMPLETED
THREAT CASE    3 COMPLETED
  .       .    .      .
  .       .    .      .
  .       .    .      .
  .       .    .      .
THREAT CASE   42 COMPLETED
THREAT CASE   43 COMPLETED
THREAT CASE   44 COMPLETED
THREAT CASE   45 COMPLETED

RANGE  STARTING EID  ENDING EID   PNP %      Aeff sq-m
   1              1       10000   96.11964   152.60001

TOTAL PRROBABILITY OF NO PENETRATION (%) =     96.11964

TOTAL EFFECTIVE AREA (SQ-M) =    152.60001

CREATE A SUPERTAB INPUT FILE FOR CONTOUR PLOTS ? <CR=YES> >NO

$
```

Figure 6.3-1. Example of BUMPER Run (Continued)

```
*********************

   BUMPER VER 4.0

Last Update 5/25/87

*********************

MAN-MADE ORBITAL DEBRIS ANALYSIS
SPACECRAFT EXPOSURE TIME (YEARS) =  10.00
OPERATING ALTITUDE (km) =  500.00
GEOMETRY OUTPUT FILE = EXAMPLE.GEM

RESPONSE OUTPUT FILE = EXAMPLE.RSP


RANGE  STARTING EID  ENDING EID   PNP %      Aeff sq-m
    1             1       10000   96.11964   152.60001

TOTAL PRROBABILITY OF NO PENETRATION (%) =     96.11964

TOTAL EFFECTIVE AREA (SQ-M) =    152.60001

$
```

Figure 6.3-2. BUMPER Summary File

Year	Altitude	$\log_{10}$ (Altitude)
1	420	2.6232
2	430	2.6335
3	445	2.6484
4	460	2.6628
5	440	2.6435
		$\Sigma =$ 13.2114

$$\text{Average} = \frac{13.2114}{5} = 2.6423$$

$$\text{Altitude} = 10^{(2.6423)} = 438.8 \text{ km}$$

Figure 6.3-3. Example of Orbital Altitude Calculation.

The next series of questions define the starting and ending element IDs for each range (SSCE). The example in figure 6.3-2 is for the entire model; therefore, the range input is from 1 to 10,000, which encompasses the entire range of element IDs. Figure 3.4-1 shows an example of a summary output file with many SSCEs defined.

The next two questions identify the appropriate data base files generated by GEOMETRY and RESPONSE. These files are in a binary format for speed of execution and compact storage.

The last question asks whether the user wishes to create a SUPERTAB-compatible file containing the probability of penetration per element surface area data for each element. Figure 5.1-2 shows an example of this file. Figure 5.1-1 shows how this data may be displayed.

6.4 CONTOUR

Figure 6.4-1 shows the typical questions asked by the CONTOUR and gives the typical responses.

The first question asks for the summary file. This file will contain the user-supplied input and the results of the analysis. With the default being "CONTOUR.SUM." Figure 6.4-2 shows an example of this file.

The second question asks for the analysis type, with the default being debris.

The third question asks for the exposure time, with a default value of 10 years.

The fourth question asks for the operating altitude. The recommended value is the same as outlined in section 6.3 for BUMPER. The default value is 500 km.

The next question asks for the element ID range for the calculations. As opposed to BUMPER, CONTOUR allows only one range. No default values are given.

The sixth question asks for the length units to be for input and output. The default is metric.

The next question asks for property ID, with the default being 1. Only one PID is allowed per analysis.

Type of wall configuration is then input, with the default being double plate. If double plate is chosen, then the penetration function is requested, with the default being the original one. The recommended function for Space Station type structure is the regression.

The next questions deal with the wall materials and minimum, maximum, and increment in thickness. No default is supplied. For

```
############################

    CONTOUR Version 2.0

    last update 6/5/87

############################

SUMMARY OUTPUT FILENAME (CR=CONTOUR.SUM) :

ANALYSIS TYPE ?
 1-DEBRIS <CR>
 2-METEOROIDS
ANSWER 1 OR 2 :

SPACECRAFT EXPOSURE TIME (YEARS) <CR=10.0> :

OPERATING ALTITUDE ( 400-500 km )  <CR=500> : 411

THE PROBABILITY OF NO PENETRATION WILL BE CALCULATED FOR A SPECIFIC RANGE
OF ELEMENT IDS INPUT THE STARTING AND ENDING ELEMENT ID FOR EACH RANGE

STARTING ELEMENT ID : 10000

ENDING ELEMENT ID   : 20000

INPUT IN METRIC OR ENGLISH UNITS  <CR>=METRIC : ENGLISH

PROPERTY ID = <CR=1> :

CONFIGURATION TYPE
    1- SINGLE PLATE
    2- DOUBLE PLATE <CR>
ANSWER (1 or 2) :

PENETRATION FUNCTION
    1-ORIGINAL <CR>
    2-PEN4
    3-REGRESSION
ANSWER (1-3) :

SHIELD MATERIAL
   1- 2024-T4
   2- 2219-T87
   3- 6061-T6
   4- 7075-T6
   5- INCONEL
SELECT MATERIAL NUMBER <CR>=1 : 3
```

Figure 6.4-1 Example of CONTOUR Run

```
MINIMUM SHIELD THICKNESS (IN) = : .01

MAXIMUM SHIELD THICKNESS (IN) = : .1

INCREMENT SHIELD THICKNESS (IN) = : .01

VESSEL WALL MATERIAL
   1- 2024-T4
   2- 2219-T87
   3- 6061-T6
   4- 7075-T6
   5- INCONEL
SELECT MATERIAL NUMBER <CR>=1 : 2

MINIMUM VESSEL WALL THICKNESS (IN) = : .075

MAXIMUM VESSEL WALL THICKNESS (IN) = : .25

INCREMENT VESSEL WALL THICKNESS (IN) = : .025

THE NUMBER OF POTENTIAL CASES IS     80
DO YOU WISH TO CONTINUE ? <CR=YES>  :

SHIELD STAND-OFF (IN) = : 4.325

INCLUDE 30 LAYERS OF MLI BETWEEN PLATES <CR>=Y :

GEOMETRY OUTPUT FILENAME ? <CR=STATION.GEM> >

   1 0.0100 4.3250 0.0750    98.11780

   2 0.0200 4.3250 0.0750    98.78834

   3 0.0300 4.3250 0.0750    99.05371
   .  .      .      .          .
   .  .      .      .          .
   .  .      .      .          .
   .  .      .      .          .
   .  .      .      .          .

  66 0.0900 4.3250 0.2500    99.83081

  67 0.1000 4.3250 0.2500    99.83659
```

Figure 6.4-1 Example of CONTOUR Run (Continued)

```
###########################

    CONTOUR Version 2.0

    last update 6/5/87

###########################

MAN-MADE ORBITAL DEBRIS ANALYSIS

SPACECRAFT EXPOSURE TIME (YEARS) =  10.00

OPERATING ALTITUDE (km) =  411.00

STARTING ELEMENT ID =   10000
ENDING ELEMENT ID   =   20000

PROPERTY ID =    1

DOUBLE PLATE ANALYSIS

ORGINAL PENETRATION FUNCTION

SHIELD MATERIAL = 6061-T6

MINIMUM MAXIMUM AND INCREMENT SHIELD THICKNESS (IN) =
 0.0100 0.1000 0.0100

VESSEL WALL MATERIAL = 2219-T87

MINIMUM MAXIMUM AND INCREMENT VESSEL WALL THICKNESS (IN) =
 0.0750 0.2500 0.0250

SHIELD STAND-OFF (IN) =      4.32500

30 LAYERS OF MULTI-LAYER INSULATION INCLUDED
GEOMETRY OUTPUT FILE = STATION.GEM

    1 0.0100 4.3250 0.0750    98.11780

    2 0.0200 4.3250 0.0750    98.78834

    3 0.0300 4.3250 0.0750    99.05371
    .    .      .      .          .
    .    .      .      .          .
    .    .      .      .          .
    .    .      .      .          .
   66 0.0900 4.3250 0.2500    99.83081

   67 0.1000 4.3250 0.2500    99.83651
```

Figure 6.4-2 Sample CONTOUR Summary File

single plate structure, only the vessel wall questions are asked.

Once all the wall parameters are input, the number of potential configurations is calculated and displayed. If the user agrees with the number the analysis continues by inputying "YES" or a <CR>. If the number is incorrect, then the wall inputs may be repeated by inputing "NO."

Next, for double plate structure, the shield standoff and whether or not to include MLI is input.

The last question asks for the geometry data base filename. This was the file created by the GEOMETRY code. The default is "STATION.GEM."

As each configuration is analyzed the results are written to both the screen and the summary file. An example of the summary file is shown in figure 6.4-2.

6.5 RPLOT

RPLOT reads the binary response data base and produces a formatted user-defined output file containing the critical diameter data for each PID and impact velocity for 0-, 15-, 30-, 45-, and 60-deg impact angles. Figure 6.5-1 shows the typical questions asked by the program. As can be seen, the questions are limited to the input and output filenames. Figure 6.5-2 shows a typical RPLOT output file.

```
RUN RPLOT

$$$$$$$$$$$$$$$$$$$$

    RPLOT VER 1.1

$$$$$$$$$$$$$$$$$$$$

RESPONSE OUTPUT FILENAME <CR=STATION.RSP> :STATION.RSP

OUTPUT FILENAME <CR=RPLOT.DAT> : RPLOT.DAT

OUTPUT LOCATED IN RPLOT.DAT
```

Figure 6.5-1 Example of RPLOT Run

```
PID       Vi        0        15        30        45        60
  1    0.2500   1.0366   1.0725   1.1973   1.4640   2.0707
  1    0.5000   0.6506   0.6731   0.7515   0.9195   1.3009
  1    0.7500   0.5035   0.5216   0.5815   0.7123   1.0092
  1    1.0000   0.4236   0.4388   0.4891   0.5990   0.8474
  .       .        .        .        .        .        .
  .       .        .        .        .        .        .
  .       .        .        .        .        .        .
  .       .        .        .        .        .        .
  1   16.2500   0.6979   0.7040   0.7235   0.7611   0.8300
  1   16.5000   0.6953   0.7013   0.7207   0.7582   0.8268
  1   16.7500   0.6927   0.6987   0.7180   0.7554   0.8237
  1   17.0000   0.6901   0.6961   0.7154   0.7526   0.8207
  2    0.2500   0.9562   0.9899   1.1039   1.3541   1.9153
  2    0.5000   0.6166   0.6382   0.7131   0.8716   1.2327
  2    0.7500   0.4818   0.4986   0.5564   0.6808   0.9628
  2    1.0000   0.4062   0.4204   0.4690   0.5738   0.8113
  .       .        .        .        .        .        .
  .       .        .        .        .        .        .
  .       .        .        .        .        .        .
  .       .        .        .        .        .        .

      16.2500   0.6243   0.6297   0.6471   0.6807   0.7424
  2   16.5000   0.6219   0.6273   0.6446   0.6782   0.7395
  2   16.7500   0.6195   0.6249   0.6422   0.6756   0.7368
  2   17.0000   0.6172   0.6226   0.6398   0.6731   0.7340
```

Figure 6.5-2 Sample RPLOT Output File

7.0 APPLICATION OF THEORY

7.1 HISTORY

In the past determining the probability of no penetration (PNP) of spacecraft subject to meteoroid impact involved the Poisson's model as described in equation 1.

$$PNP = e^{-FAT} \tag{1}$$

where PNP is the probability of no penetration, F is the flux of the design particle, A is the spacecraft area, and T is the exposure time.

Previously, because spacecraft had small areas and short exposure times, the shielding required using this analysis was not significant. Also, the effect of man-made orbital debris was not previously considered.

When using the preceding analysis for the design of the Space Station, the required shielding for man-made debris became excessive. Exact meanings of the terms in this equation were also confusing. What area is used, surface, projected, effective, or some factor of either? How is a design particle selected?

This lead to the development of a more rigorous technique, outlined in reference 1, appendix G. The benefit of this more detailed analysis is that it accounts for the actual geometry of the spacecraft, its orientation, and all of the impact characteristics. This leads to more acceptable shielding requirements in terms of weight without compromising the required reliability of the spacecraft.

7.2 DETERMINING THE PROBABILITY OF NO PENETRATION

The analysis technique in reference 1, appendix G breaks the spacecraft into a series of flat plate elements much like a FEM. The threat environment (both debris and meteoroids) is also broken into finite cases, each for which a probability can be assigned. Each element is then evaluated for each threat. The PNP for a specific element is given by

$$PNP = EXP\left[- \left[A * T * \sum_{i=1}^{NT} F(D_c(v_i, \beta_i)) * \cos(\beta_i) * \rho(\alpha_i, \theta_i) * I_s\right] \right] \tag{2}$$

where A is the elements surface area, T is the exposure time, NT is the number of threats, β is the angle between the plates normal and the threat vector (the impact angle), $p(\alpha_i, \theta_i)$ is the probability of the threat occurring, $F(Dc(V_i, \beta_{ij}))$ is the flux of the particle that penetrates the plate for the given impact velocity and impact angle, and Is is a indicator equal to 1 if the element is exposed to the threat and 0 if it is not.

To determine the PNP for a range of elements the following equation is used

$$PNP = EXP\left[- \left[T * \sum_{j=1}^{NELM} A_j * \sum_{i=1}^{NT} F(D_c(v_{ij}, \beta_{ij})) * \cos(\beta_{ij}) * \rho(\alpha_i, \theta_i) * l_s\right]\right] \quad (3)$$

where NELM is the number of elements.

The preceding equations show that several items of data are required for each element. These data can be broken into logical groups. The first group consists the data specific to the threat and the spacecraft geometry. This consists of the threat data (impact velocity and the probability of the threat occurring) and the spacecraft interaction with the threat (is the element exposed and the cosine of the impact angle). These data are calculated by the GEOMETRY computer program.

The second group consists of data specific to the penetration resistance of the spacecraft wall (the diameter that just penetrates for specific impact conditions). These data are calculated by the computer program RESPONSE.

This information is brought together along with the exposure time and element range of interest to perform the PNP calculation in the BUMPER computer program.

Breaking the PNP calculation into several distinct parts has many advantages. The most expensive part of the analysis, in terms of computer runtime, is creating the threat and spacecraft geometry interaction data base. This data base need only be created once for each spacecraft geometry and orientation. Trade studies involving changes in the wall configuration or spacecraft exposure time then use this common data base. Breaking down the analysis also produces modular computer codes that are more adaptable to modifications and future additions.

A detailed discussion of each of these computer codes follows:

7.3 GEOMETRY VERSION 2.51

GEOMETRY produces the geometry data base used by the BUMPER code. Specifically, it produces information about the threat (polar angles, impact velocity and probability), the finite elements in the model (element ID, property ID, surface area), and the interaction between the model and the threat (list of exposed elements and their impact angle cosines for each threat). The information is stored in a user-defined binary file.

The environment is modeled as a series of distinct threat cases as outlined in section 7 of reference 1. The spacecraft geometry is input through the use of a FEM. The model is limited to triangular elements whose nodes must be in the same coordinate system. In addition, the X-axis of the this coordinate system must be parallel to the spacecraft's velocity vector and the Z-axis must be parallel to the Earth radius vector pointing away from the Earth. This is shown in figure 2.2-1. The model information must be stored in a file using the SUPERTAB Universal File Format. Other formats could be used but would require modification of the code.

Figure 7.3-1 shows a simplified flow chart of the computer program. Each line item in the figure will be discussed in more detail in the following sections, but it should be noted that no attempt will be made to describe the code in complete detail. The code is well documented internally, and appendix B contains a complete listing.

The code consists of two main areas, as shown in figure 7.3-1. The first consists of the global calculations, which need only be calculated once. The remainder of the code is specific to determining which elements are exposed to a given threat. This portion of the code can be further broken down into two areas, the first being the elimination of the elements that do not face the threat (back side elements) and the second being the elimination of elements shielded from the threat by other elements.

7.3.1 Write Header

This portion of the code writes the program header to the screen. It also reads in the analysis type (meteoroid or debris), the name of the file containing the FEM, and the name of the output file.

7.3.2 Calculate the Threat Data

The threat data for meteoroids or debris is calculated here. These data consist of the polar angles describing the threat direction, the impact velocity associated with the threat, and the probability of the threat occurring. The user specifies the number of threats to be evaluated.

The debris threat is defined by JSC 20001. The code uses the relative velocity distribution and derives from it the threat angle distribution. The velocity distribution is contained in an external file DEB.VEL. The format for the velocity distribution file is contained in the code listing. The relationship between relative velocity and threat angle are shown graphically in figure 7.3-2.

The meteoroid threat is defined by NASA SP-8013. For simplification, a constant inertial velocity of 20 km/s for all meteoroids is assumed. An isotropic threat is also assumed; therefore, all directions have equal likelihood of occurring. The spacecraft motion is accounted for by increasing or decreasing the probability appropriately (see sec. 2 of ref. 1). Earth shadowing is accounted for by ignoring the directions that lay below the horizon. For meteoroid analysis, a 500 km spacecraft altitude is assumed. For a more detailed discussion of the threat, see sections 2 and 7 of reference 1.

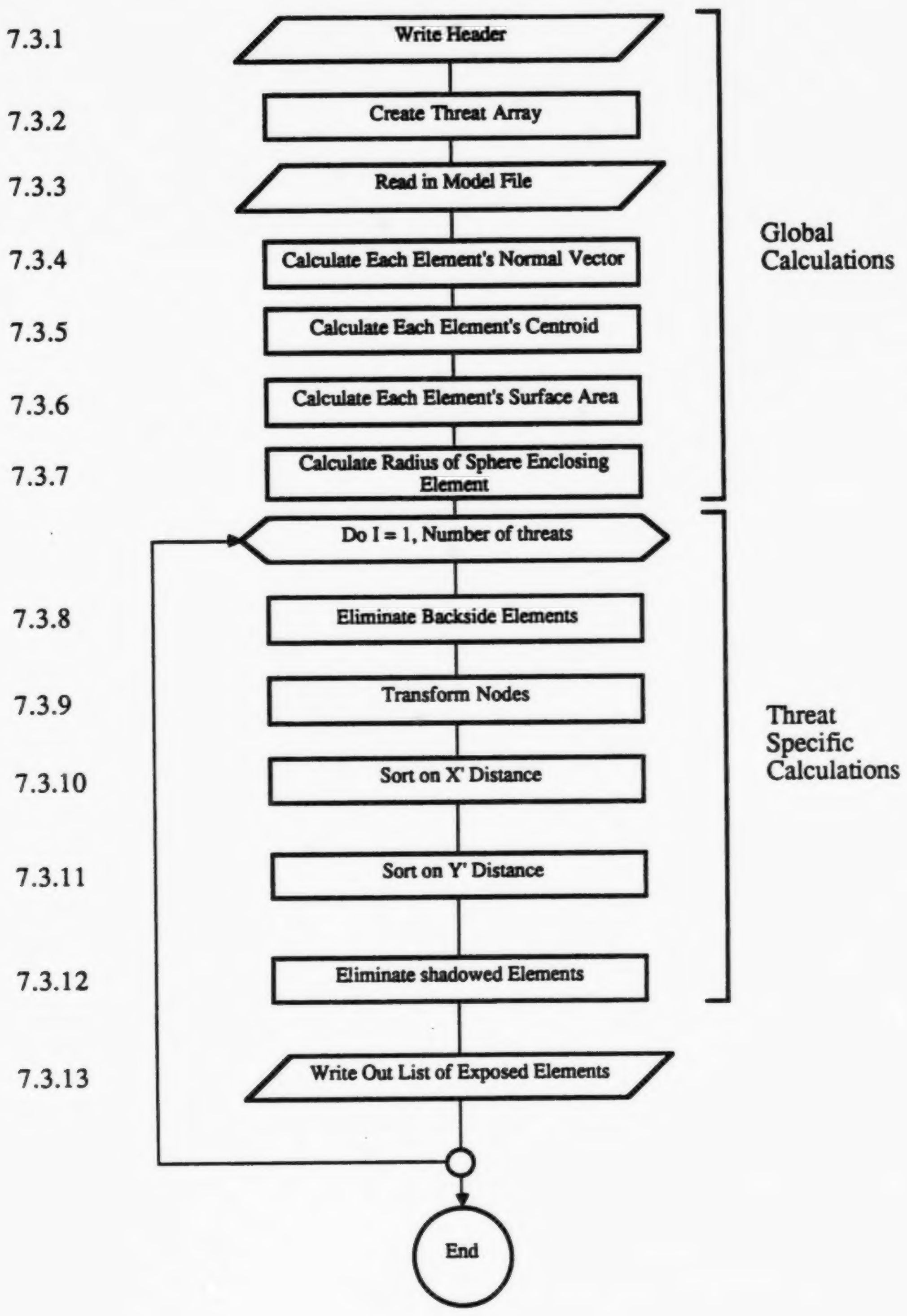
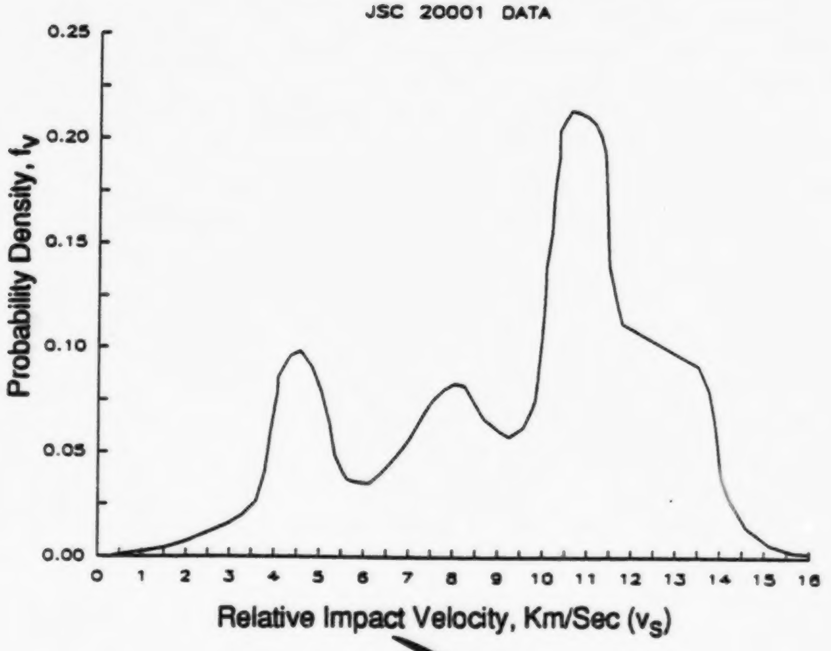

7.3.1
Write Header
7.3.2
Create Threat Array
7.3.3
Read in Model File
7.3.4
Calculate Each Element's Normal Vector
7.3.5
Calculate Each Element's Centroid
7.3.6
Calculate Each Element's Surface Area
7.3.7
Calculate Radius of Sphere Enclosing Element
Global Calculations
Do I = 1, Number of threats
7.3.8
Eliminate Backside Elements
7.3.9
Transform Nodes
7.3.10
Sort on X' Distance
7.3.11
Sort on Y' Distance
7.3.12
Eliminate shadowed Elements
Threat Specific Calculations
7.3.13
Write Out List of Exposed Elements
End

Figure7.3-1. GEOMETRY Flow Chart (Version 2.0)



$$f_\theta (v) = | -2v_S \sin\theta | \, fv (2v_S \cos\theta)$$

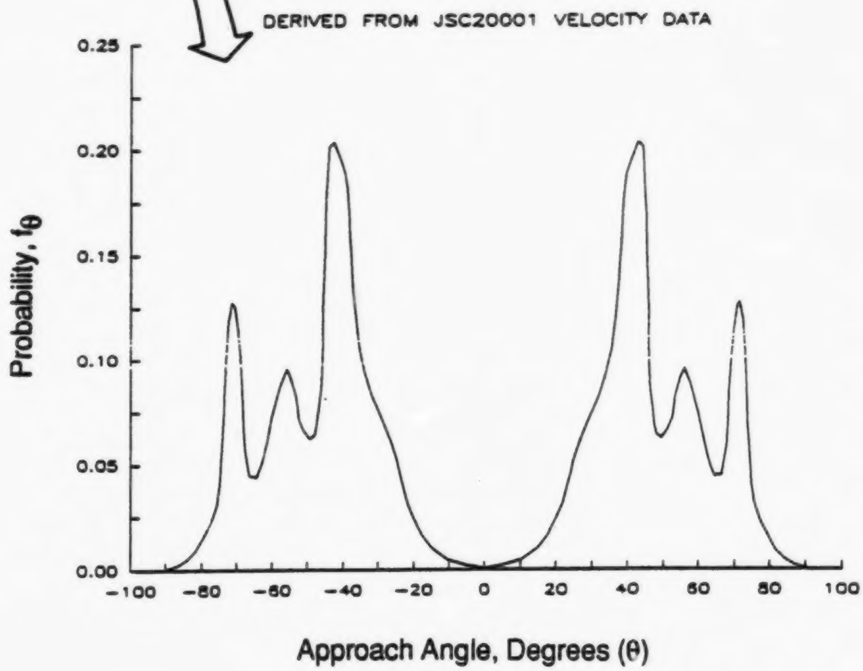


Figure 7.3-2. Transformation of Velocity Distribution to Approach Angle Distribution

7.3.3 Read in the Model File

The model file is read in and stored in global arrays for further use. One benefit of the code is the use of common blocks contained in external files, allowing easy modification of memory requirements.

7.3.4 Calculate the Normal Vectors

Each element's normal vector is calculated in the global coordinate system. The model builder must take care to ensure the correct orientation. The code uses nodes in order 1, 2, 3 and the right-hand rule to assign the normal vector direction.

7.3.5 Calculate the Centroid Locations

Each element's centroid is calculated in the global coordinate system by averaging the three nodal locations.

7.3.6 Calculate the Surface Areas

Each element's surface area is calculated and saved for output.

7.3.7 Calculate the Radius of Sphere Enclosing the Element

The radius of the sphere that just encloses each element is calculated and saved. This information is used later to evaluate the elements for shadowing.

7.3.8 Eliminate Back Side Elements

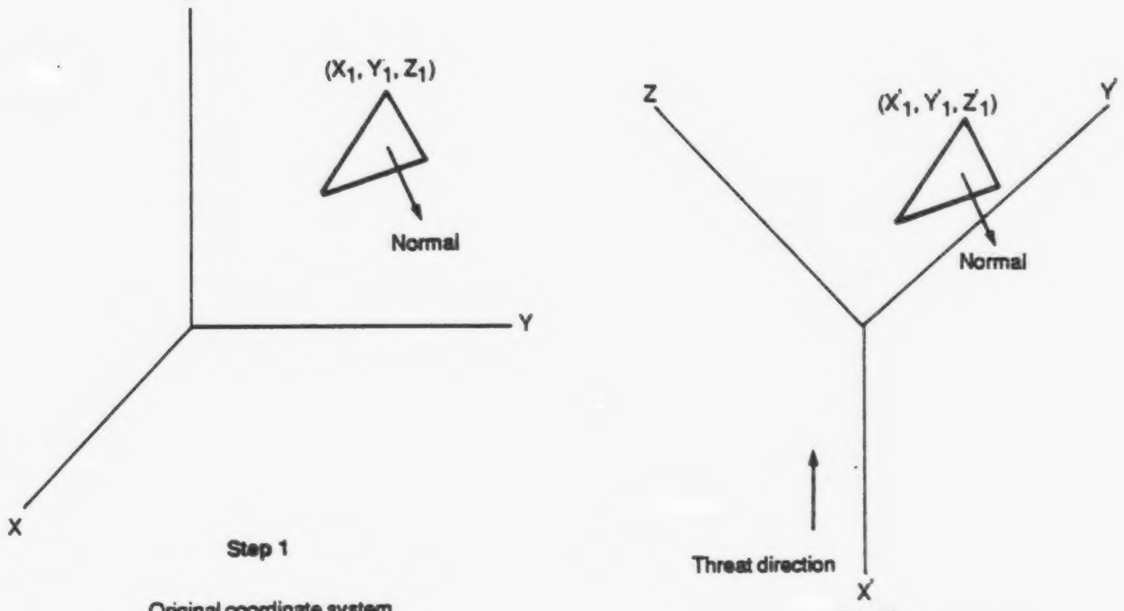
For each threat the elements having normal vectors pointing away from the threat are eliminated to prevent impacts on the element back faces. This is done by calculating the cosine of the angle between the element's normal vector and a vector pointing toward the threat direction. This is also the cosine of the impact angle and is saved for output later if the element is exposed. Those elements with negative cosines are termed "back side elements" and are removed from further consideration for that specific threat.

7.3.9 Transform the Nodal Coordinates and Centroid Locations

Those elements not eliminated as back side elements now have their nodal coordinates and centroid locations transformed into a coordinate system where the X-axis is parallel to the threat direction. The coordinates are then projected onto a plane orthogonal to the threat direction located at the centroid of the element. This is shown in figure 7.3-3.

7.3.10 Sort on the Transformed X-Dimension

The potentially exposed elements (those not eliminated as back side) are sorted in descending order on the distance from the origin to the elements centroid in the transformed X-dimension shown in figure 7.3-3. The quick-sort algorithm is used.

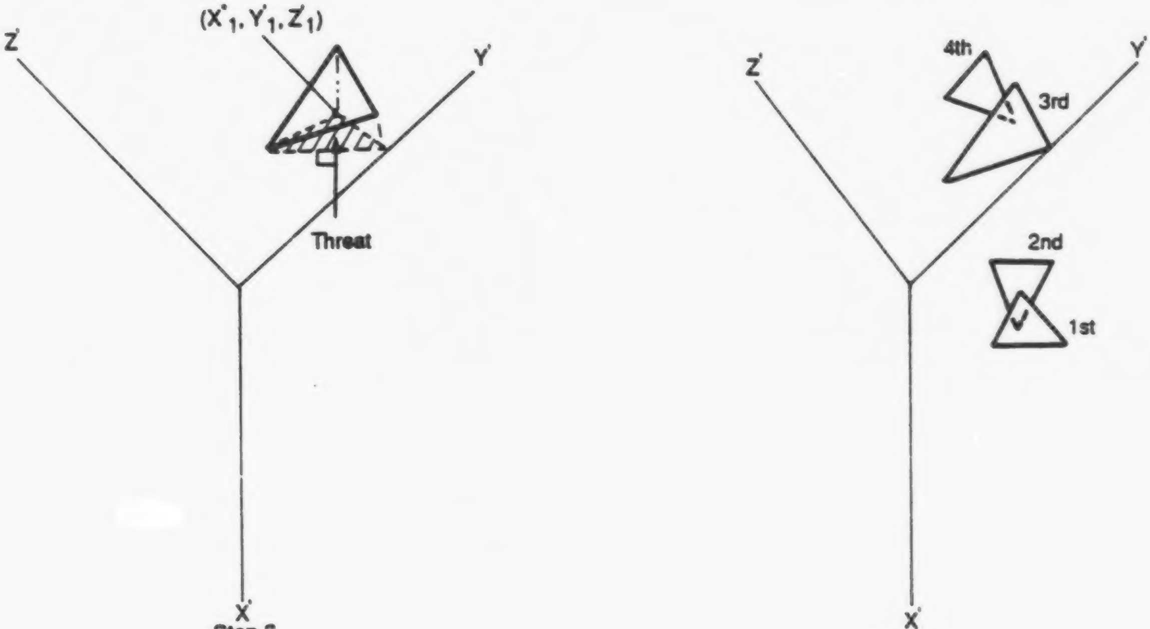

Z
(X_1, Y_1, Z_1)
Normal
Y
X
Z
(X'_1, Y'_1, Z'_1)
Y'
Normal
Threat direction
X'

Step 1

Original coordinate system

Step 2

Transformed coordinate system - X' parallel to threat direction


(X''_1, Y'_1, Z'_1)
Z'
Y'
Threat
X'
4th
3rd
2nd
1st
Z'
Y'
X'

Step 3

Project coordinates on to plane orthagonal to threat

Step 4

Sort elements in X' then Y' for efficient use of hidden surface algorithm

Figure 7.3-3. Transformation of Nodes and Sorting of Elements

7.3.11 Sort on the Transformed Y-Dimension

The potentially exposed elements are sorted in descending order on the distance from the origin to the element's centroid in the transformed Y-dimension shown in figure 7.3-3. The quick-sort algorithm again is used.

7.3.12 Eliminate the Shadowed Elements

An element is in the shadow of another element only if its centroid lies behind the shadowing element and within the projected boundaries of the shadowing element. This is done with the sorted lists of elements in the transformed X- and Y-dimensions. The potentially shadowed element must lie below the shadowing element in the sorted transformed X-list. Only those elements whose centroids lay within a distance of the radius enclosing the shadowing element in the transformed Y-dimension are considered. This is done using the sorted transformed y list. If an element passes this test, a more accurate test is performed to determine if it lies within the projected boundaries. This portion of the code is complex and is best understood by following an example through the shadowing subroutine. This will not be done in this document, but the subroutine is well documented internally.

7.3.13 Output List of Exposed Elements and Associated Impact Angles

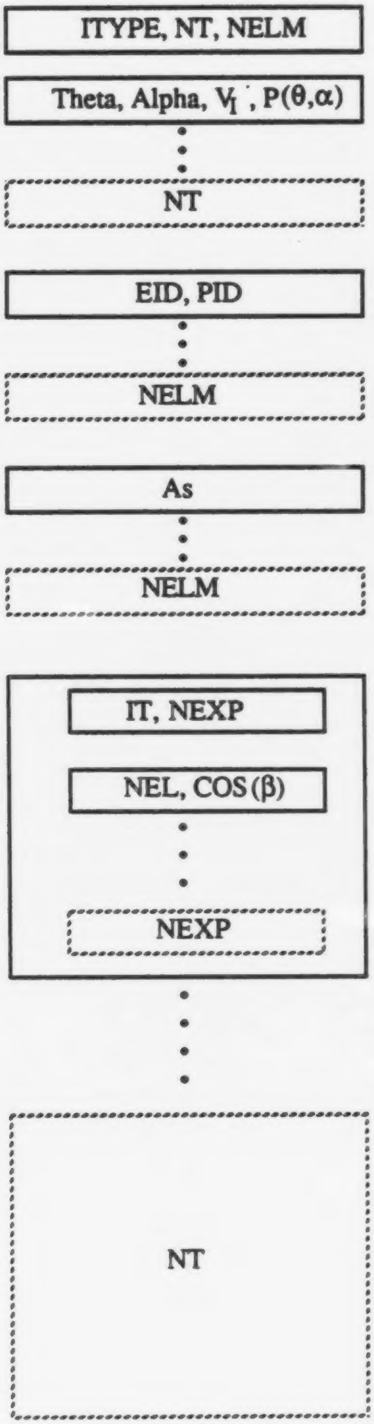
For each threat case a list of the exposed elements and their associated impact angle cosines is output. Additional outputs are the analysis type, the threat data, element surface area, element ID, and property ID. All this information is contained in binary format in the user-specified output file. Figure 7.3-4 shows the structure of this file.

7.4 RESPONSE VERSION 2.0

RESPONSE generates the critical diameter lookup tables for use by the BUMPER code. These tables list the critical diameter as a function of property ID (each finite element has been assigned one when the model was built), impact angle, and impact velocity. The information is saved in a user-defined file in binary format. The code is limited to aluminum structures, and specifically for double plate structures the shield thickness must be between 10% and 50% of the vessel wall thickness.

A simplified flow chart of the code is show in figure 7.4-1. Each line item in the flow chart will be discussed in more detail in the following sections, but no attempt will be made to explain all the details of the code. The source code is internally documented, and a full listing of is contained in appendix C.

The code can be broken into two parts for discussion purposes. The first part consists of reading in the user-inputs. These consist of the type of penetration function to be used and various dimensions and materials. The second part deals with calculating the critical diameters (the diameter that just penetrates) as a function


ITYPE, NT, NELM
Theta, Alpha, V_I, P(θ,α)
NT
EID, PID
NELM
As
NELM
IT, NEXP
NEL, COS (β)
NEXP
NT

Integer*2

IT = Threat case
ITYPE = Analysis type (1 - Deb, 2 - Met)
NT = Number of threat cases
NELM = Number of elements
NEL = Element number
NEXP = Number of exposed elements for a particular threat case

Integer*4

EID = Element ID
PID = Element Property ID

Real*4

As = Surface Area, m^2
COS (β) = Cosine of impact angle
P (θ,α) = Probability of threat occurring
V_I = Impact velocity for specific threat, km/s
Theta, Alpha = Polar angles describing threat

Figure 7.3-4. Geometry Data Base File Structure.

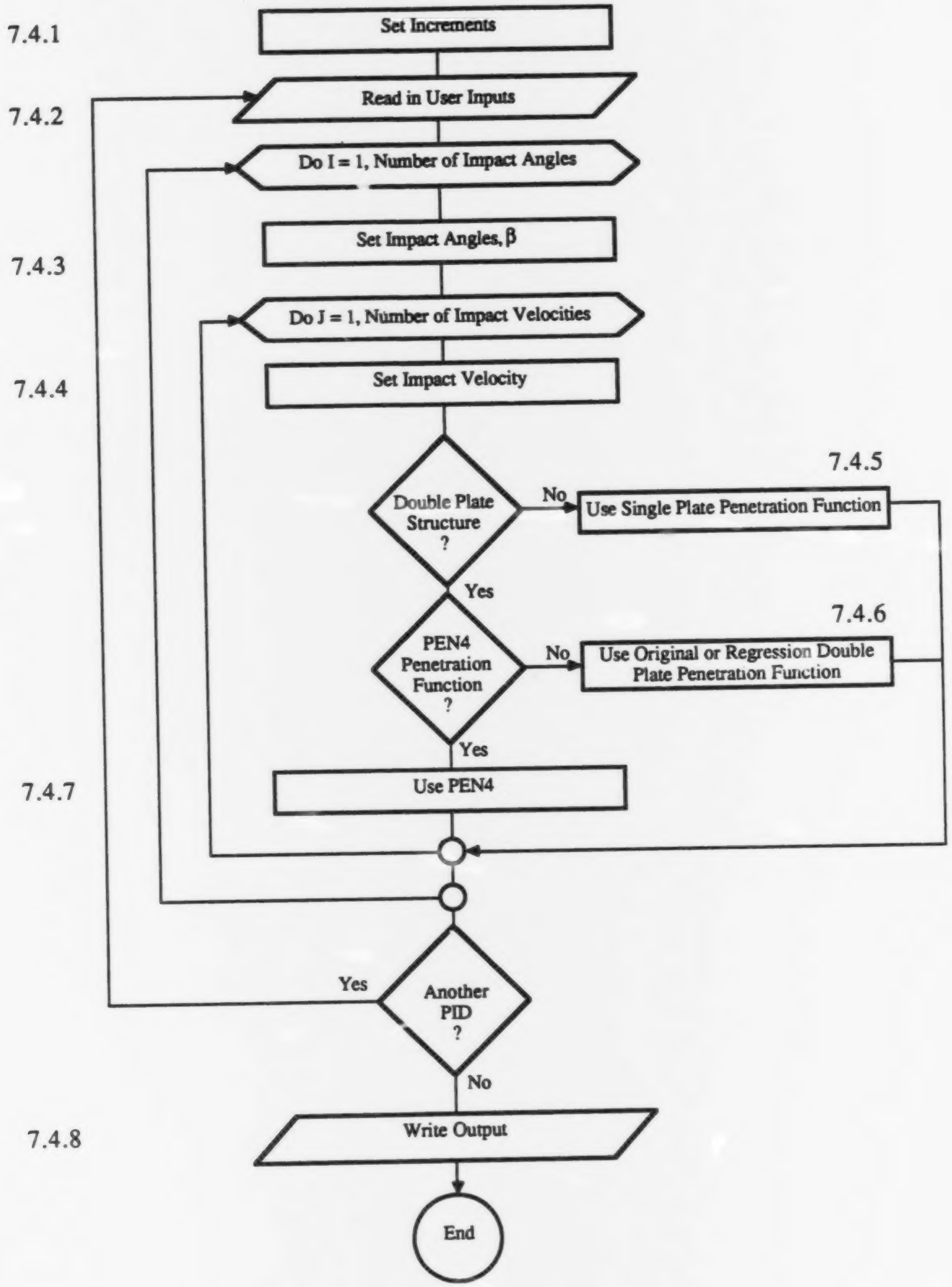

7.4.1
Set Increments
7.4.2
Read in User Inputs
Do I = 1, Number of Impact Angles
Set Impact Angles, β
7.4.3
Do J = 1, Number of Impact Velocities
7.4.4
Set Impact Velocity
Double Plate Structure ?
No
7.4.5
Use Single Plate Penetration Function
Yes
PEN4 Penetration Function ?
No
7.4.6
Use Original or Regression Double Plate Penetration Function
Yes
7.4.7
Use PEN4
Yes
Another PID ?
No
7.4.8
Write Output
End

Figure 7.4-1. RESPONSE Flow Chart (Main Program.)

of impact angle and impact velocity. The results are output in the user-defined file.

7.4.1 Set Increments

The increments for impact angle and impact velocity are explicitly set in the code. For meteoroid analysis the impact angle is varied in 5-deg increments and the impact velocity in 1 km/s increments with a maximum impact velocity of 70 km/s. For man-made debris the impact angle is also varied in 5-deg increments, while the impact velocity is varied in 0.25 km/s increments with a maximum impact velocity of 17 km/s. These values can be changed but would require modification of the source code.

7.4.2 Read in User Inputs

The user-specified inputs are read in at this point. For the initial property ID they consist of the analysis type, output file names, and the type of input units (English or metric). Additionally, the type of wall configuration (single or double plate) is read in. For double plate structures three optional penetration functions are available. The user also specifies the plate thicknesses, the shield standoff, and whether or not multilayer insulation is included. For all additional property IDs the default values are those input for the previous property ID.

This portion of the code also reads in the material property file MAT.PRP. This file contains the physical property data for several common aluminum alloys. Additional aluminum alloys can be added. The format of the file is shown in figure 7.4-2.

7.4.3 Set the Impact Angle

The code sets the impact angle using the increments defined in section 7.4.1. Because data is limited for impact angles greater than 60 deg, the code considers angles greater than 60 deg to be equivalent to 60 deg.

7.4.4 Set the Impact Velocity

The impact velocity is set using the previously defined increments given in section 7.4.1.

7.4.5 Single Plate Penetration Function

The response of single plates subject to meteoroid or debris impact is modeled with the Schmidt-Holsapple crater volume equation, (sec. 4 and 7 of ref. 1). The effect of multilayer insulation is ignored. In addition spall is addressed by defining failure as hemispherical crater depths exceeding 70% of the plate thickness. A typical response curve is shown in figure 7.4-3.

7.4.6 Double Plate Penetration Function

As stated previously, the user has the option of three unique penetration functions for double plate structures. However, once one

Figure 7.4-2. Example of MAT.PRP File (Material Property File.)

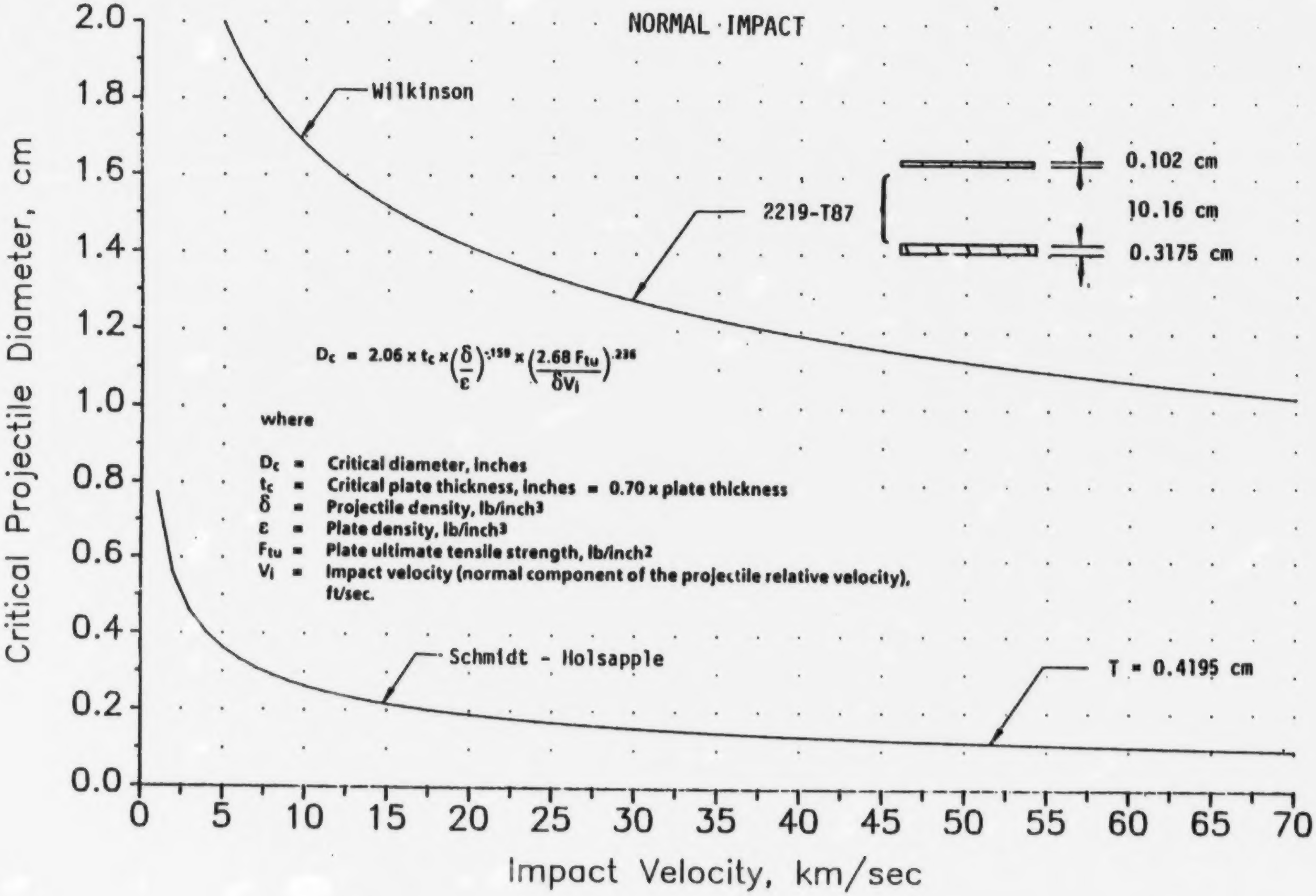
Material Name	Density (lb/in^3)	Tensile Yield Strength (psi)	Tensile Ultimate Strength (psi)	Shear Strength (psi)	Wilkinson's[1] Constant (km/s)	Speed of Sound (ft/s)	Shock[2] Projectile Velocity (unitless)	Brinnel Hardness
2024-T4	.100	37000.	54000.	29000.	.425	17024.	1.345	120.
2219-T87	.102	52000.	63000.	37000.	.270	16620.	1.345	130.
6061-T6	.098	27000.	42000.	35000.	.292	16630.	1.345	73.
7075-T6	.101	68000.	78000.	47000.	.345	16540.	1.345	150.

Fortran Format (A12, 8E12.5)

References:

[1] "A Penetration Criterion For Double-Wall Structures Subject To Meteoroid Impact," J.P.D. Wilkinson, AIAA Journal, October 1969.

[2] Empirical Material Constant for Aluminum.



$$D_c = 2.06 \times t_c \times \left(\frac{\delta}{\varepsilon}\right)^{-.159} \times \left(\frac{2.68\,F_{tu}}{\delta V_i}\right)^{.236}$$

where

D_c = Critical diameter, inches
t_c = Critical plate thickness, inches = 0.70 x plate thickness
δ = Projectile density, lb/inch³
ε = Plate density, lb/inch³
F_{tu} = Plate ultimate tensile strength, lb/inch²
V_i = Impact velocity (normal component of the projectile relative velocity), ft/sec.

Figure 7.4-3. Comparison of Penetration Resistance to Meteoroids of Single Wall and Double Wall Structures

has been chosen, all following property IDs in that analysis session will use the same penetration function.

The first penetration function is titled the original. This is the function with which the majority of the contract work was performed. It consists of the ballistic portion of the original PEN4, the Burch modified Bristow equations (ref. 1) for the fragmenting regime, and the Wilkinson equations for the high-velocity regime. A typical penetration function using this approach is shown in figure 7.4-4. It should be noted that the transition between the various equations is only a function of where the equations intercept and not based on physical data.

The second penetration function is the updated version of PEN4 and is discussed below in section 7.4.7.

The third penetration function is titled regression. It uses the updated version of PEN4 for the ballistic regime, a regression equation for the fragmenting regime, and Wilkinson's method for the high-velocity regime. The regression equation represents the best curve fit of the test data developed under the contract. For a more detailed discussion, see ref. 1. The regression penetration function is recommended for use in Space Station type analysis. Figure 7.4-5 shows a typical penetration function using this technique.

7.4.7 PEN4

PEN4 is the Boeing-developed hypervelocity penetration computer program. Developed by the Boeing Aerospace hypervelocity impact group, it is intended to analyze impacts in the range of 0 to 8 km/s. It has been updated to more closely predict the penetration resistance of aluminum double plate structure impacted by aluminum projectiles. A typical penetration function is shown in figure 7.4-6. A more detailed discussion of PEN4 is contained in ref. 1. It should be noted that the code does not address the effect of multilayer insulation, and the results for angles other then normal impact are considered to be suspect.

7.4.8 Write Output

The response lookup tables are written in binary format to the user-defined output file. Additionally, a summary list of information on each property ID case run is written to the user-defined summary file. Figure 7.4-7 shows the structure of this file.

7.5 BUMPER VERSION 4.0

BUMPER determines the PNP of a spacecraft subject to man-made debris or meteoroid impact. The code also determines the effective area of the spacecraft. The code can perform these calculations for all elements in the spacecraft model together or for specific ranges of element IDs. Figure 7.5-1 shows a simplified flow chart of the computer code. Each line item will be discussed in further detail in the following sections, but no attempt will be made here to explain

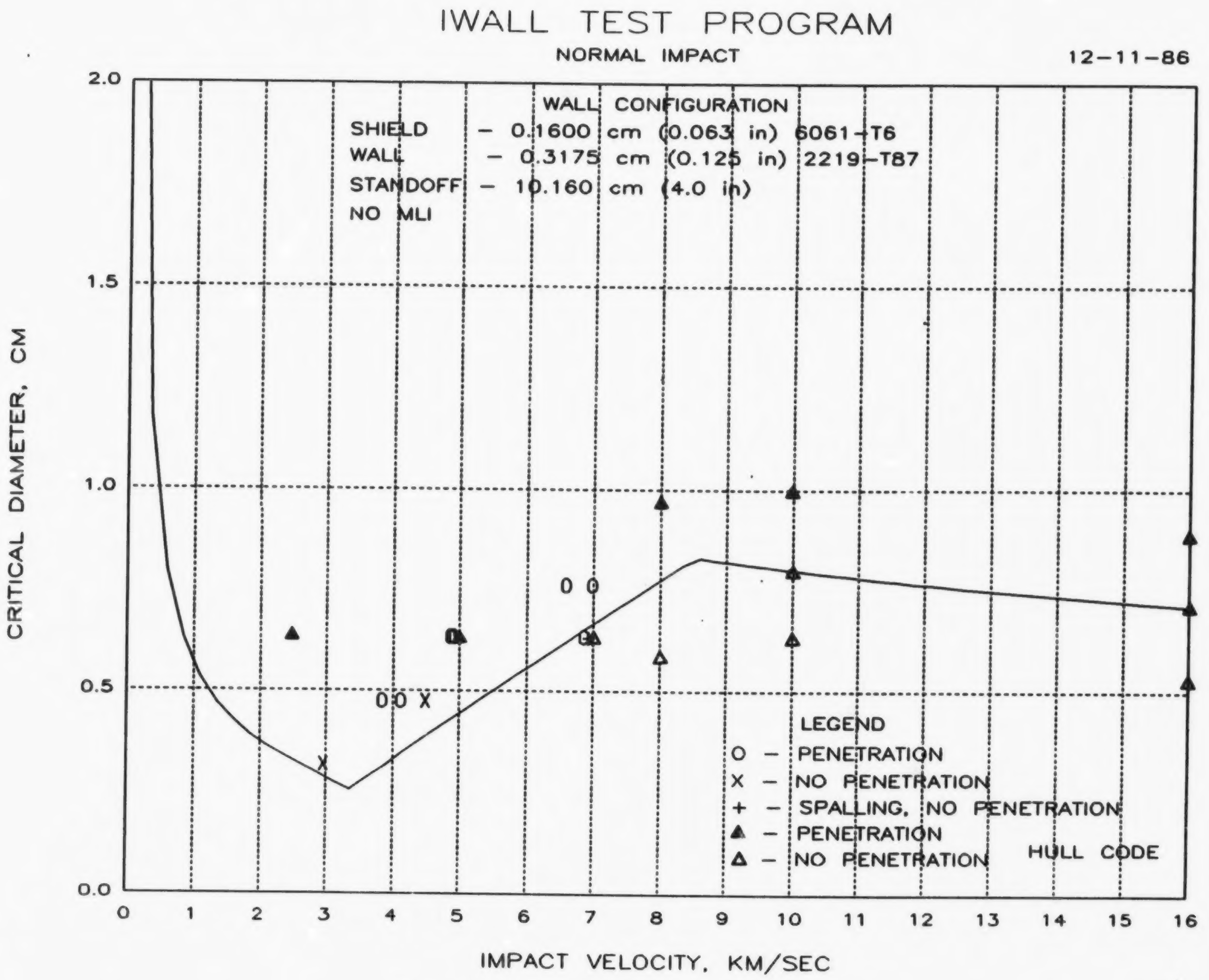


Figure 7.4-4. Penetration Function With Burch Equations

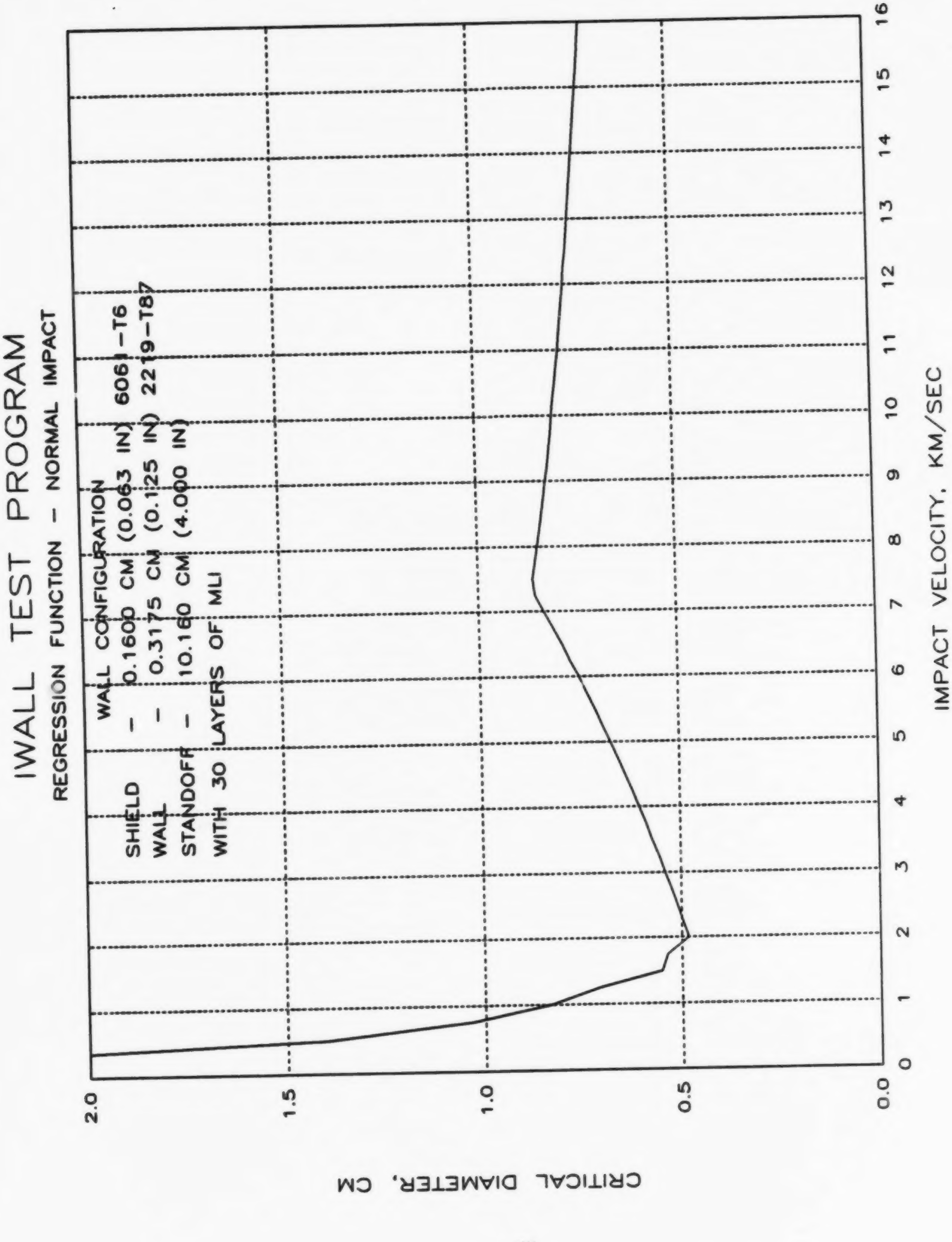
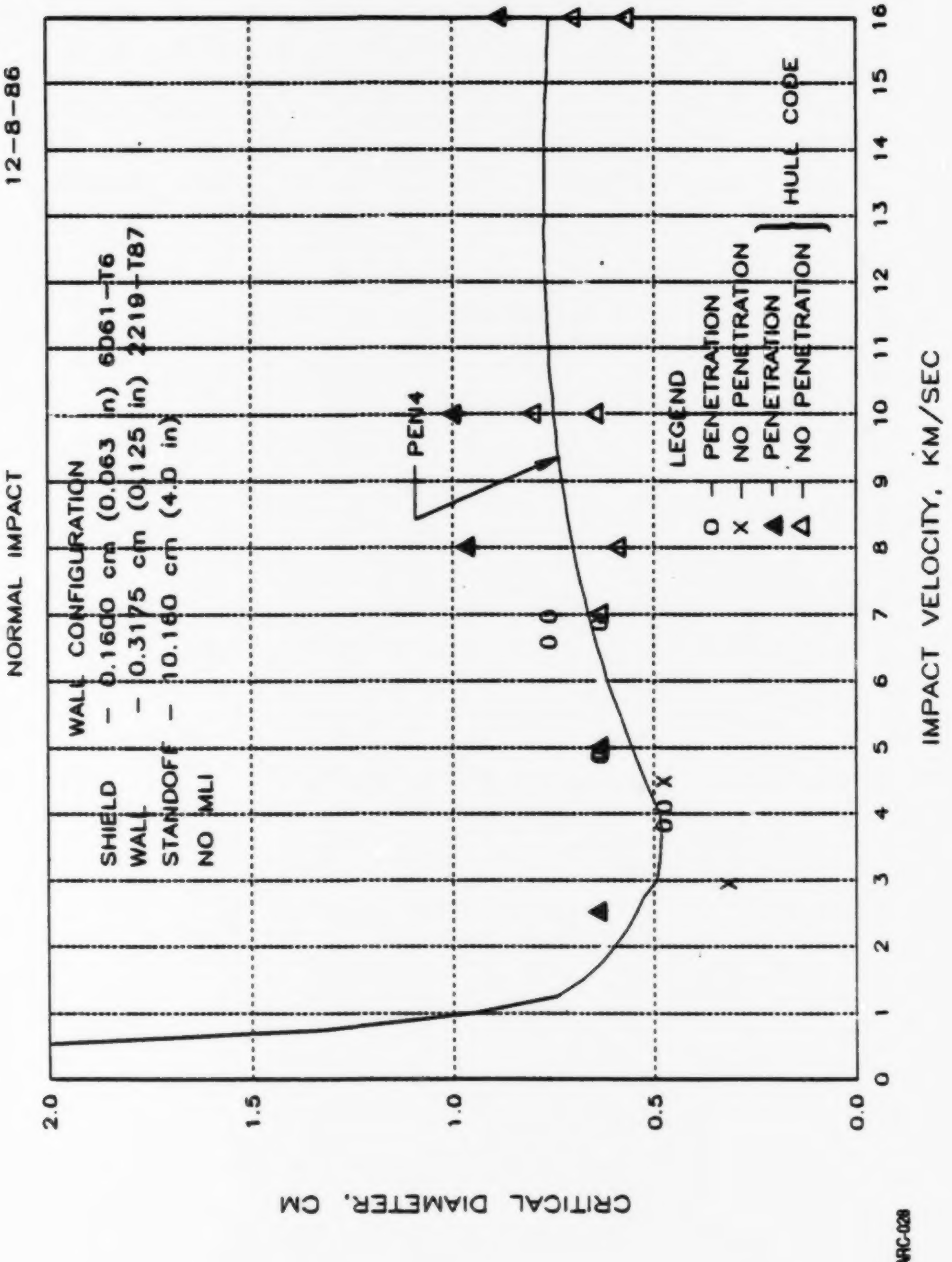


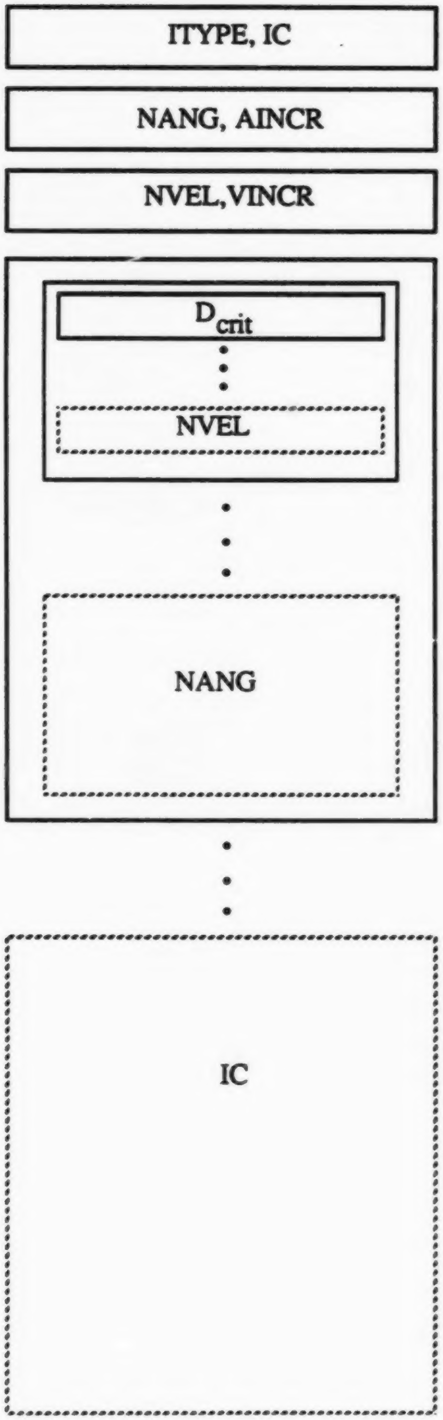
Figure 7.4-5. Regression Penetration Function



2-25-87

QRS/ARC-028

Figure 7.4-6. Penetration Function With PEN4


ITYPE, IC
NANG, AINCR
NVEL,VINCR
D_{crit}
NVEL
NANG
IC

Integer*2

IC = Number of wall configurations
ITYPE = Analysis type
NANG = Number of impact angles
NVEL = Number of impact velocities

Real*4

AINCR = Impact angle increment, radians
D_{crit} = Critical projectile diameter, cm
VINCR = Impact velocity increment, km/s

Figure 7.4-7. Response Data Base File Structure.

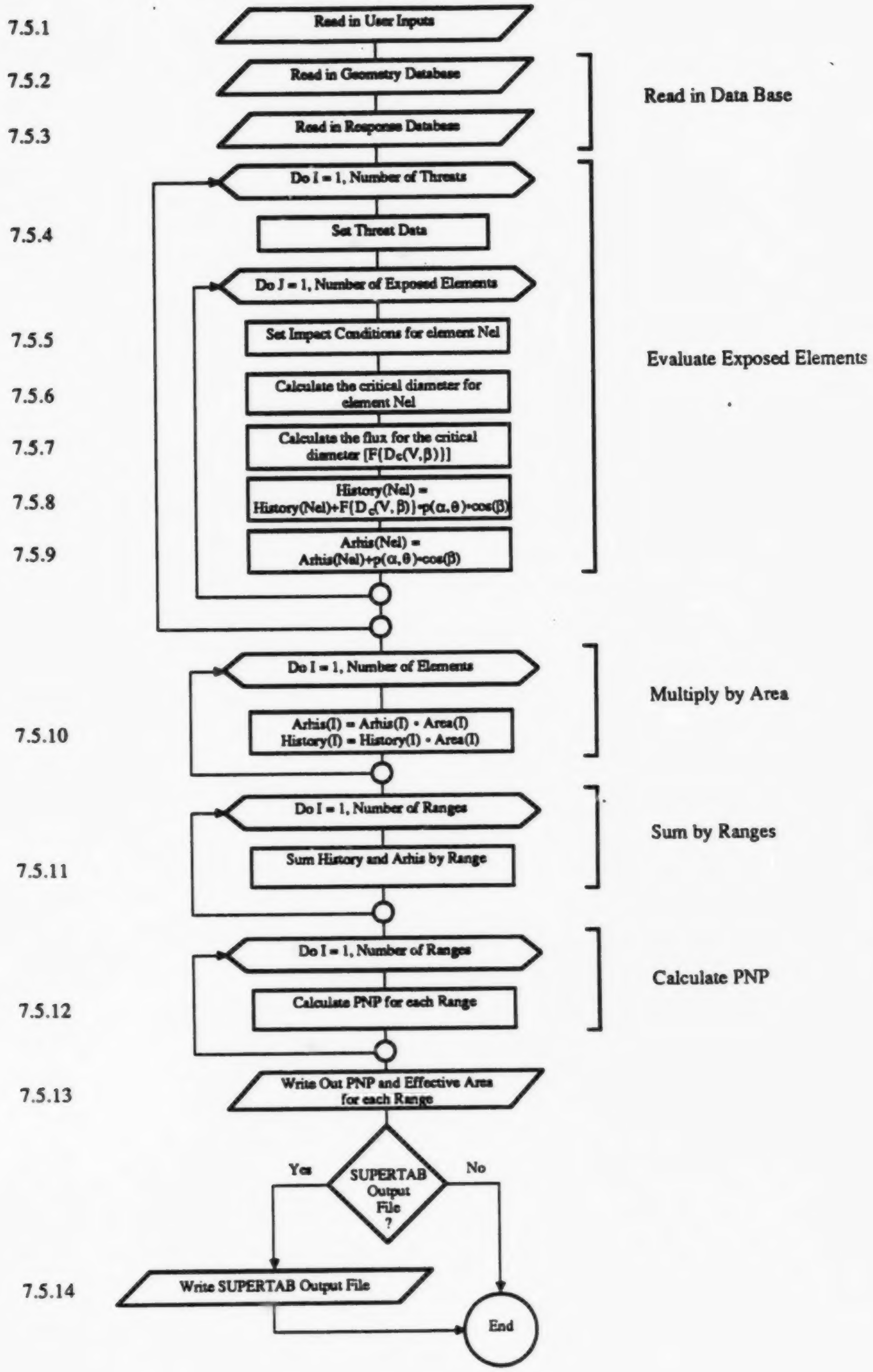

7.5.1
Read in User Inputs
7.5.2
Read in Geometry Database
7.5.3
Read in Response Database
Read in Data Base
Do I = 1, Number of Threats
7.5.4
Set Threat Data
Do J = 1, Number of Exposed Elements
7.5.5
Set Impact Conditions for element Nel
7.5.6
Calculate the critical diameter for element Nel
7.5.7
Calculate the flux for the critical diameter [F(Dc(V,β))]
7.5.8
History(Nel) = History(Nel)+F(Dc(V,β))·p(α,θ)·cos(β)
7.5.9
Arhis(Nel) = Arhis(Nel)+p(α,θ)·cos(β)
Evaluate Exposed Elements
Do I = 1, Number of Elements
7.5.10
Arhis(I) = Arhis(I) · Area(I)
History(I) = History(I) · Area(I)
Multiply by Area
Do I = 1, Number of Ranges
7.5.11
Sum History and Arhis by Range
Sum by Ranges
Do I = 1, Number of Ranges
7.5.12
Calculate PNP for each Range
Calculate PNP
7.5.13
Write Out PNP and Effective Area for each Range
SUPERTAB Output File ?
Yes
No
7.5.14
Write SUPERTAB Output File
End

Figure 7.5-1. BUMPER Flow Chart.

the code in complete detail. The code is well documented internally, and a complete listing is contained in appendix D.

As shown in figure 7.5-1, the code can be broken into five areas for discussion purposes. The first consists of reading in the geometry and response data bases. Next, the finite threat cases are looped through, evaluating only the exposed elements for each one. This consists of determining the flux of the critical projectile and storing its sum in a global array. This global array is then multiplied by the element surface area array. The array is then summed up by element ID ranges. Finally, the PNP for each range is calculated.

7.5.1 Read User Inputs

The user inputs consist of the analysis to be performed (meteoroid or debris), the summary filename, the spacecraft exposure time, the spacecraft operating altitude, and the element ID ranges. This portion of the code also writes the screen header. All user input is output to the summary file.

7.5.2 Read in the Geometry Data Base

The geometry data base created by the GEOMETRY code and stored in a binary file is read in. The data base contains the information on the threat, each element's element identification number (EID) and PID, the element's surface area, and a list of the exposed elements along with their impact angle cosine for each threat.

7.5.3 Read in the Response Data Base

The wall penetration resistance data base created by the RESPONSE code and stored in a binary file is read. The data base contains the critical diameter lookup tables for each property ID as a function of impact velocity and impact angle.

7.5.4 Set the Threat Data

For each threat the probability of the threat occurring and the impact velocity is recovered from the geometry data base.

7.5.5 Set the Impact Conditions for Element Nel

For each threat there is a list of exposed elements in the geometry data base. Nel is a specific element exposed to a specific threat. The code loops through all exposed elements for all threats. The impact conditions consist of the impact angle cosine and the property ID for the element as well as the previously recovered impact veleocity.

7.5.6 Calculate the Critical Diameter

The critical diameter for element Nel for the specific threat can now be calculated. The response data base is used through linear interpretation to determine the critical diameter. Using a lookup table approach as opposed to calculating the diameter directly allows a faster, more efficient code.

7.5.7 Calculate the Flux of the Critical Diameter

The flux of the critical diameter is calculated using NASA-specified flux equations. The meteoroid flux equations are found in JSC 30000. Gravity focusing of meteoroids is accounted for. Earth shadowing has already been accounted for in the GEOMETRY code. The man-made debris flux equations are found in JSC 20001. The log of the debris flux is assumed to vary linearly between 400 and 500 km. The flux given by JSC 20001 is multiplied by 4.0 to account for the difference in the Boeing and NASA definition of flux. No corrections are needed for the meteoroid flux. For a more detailed discussion, see appendix G of ref. 1.

7.5.8 $\mathrm{HISTORY(NEL)} = \mathrm{HISTORY(NEL)} + F(D_c(v_i,\beta_i)) * \mathrm{COS}(\beta_i) * \rho(\alpha_i,\theta_i)$

The product of the flux, the threat probability, and the cosine of the impact angle for each element is summed in the History array for all elements. When all threats have been evaluated the sum in the History array for each element represents the summation part of equation 1.

7.5.9 $\mathrm{ARHIS(NEL)} = \mathrm{ARHIS(NEL)} + \mathrm{COS}(\beta_i) * \rho(\alpha_i,\theta_i)$

The product of the threat probability and the cosine of the impact angle for each element is summed in the Arhis array for all threats. When all threats have been evaluated, this represents the effective area divided by the surface area for element Nel.

7.5.10 HISTORY(NEL) = HISTORY(NEL) * Area(NEL), ARHIS(NEL) = ARHIS(NEL) * Area(NEL)

Each element in the History and Arhis arrays is multiplied by its surface area. Each term in the History array now represents the average number of penetrating particles per unit time for a specific element. Each term in the Arhis array represents the effective area of a specific element.

7.5.11 Sum up the History and the Arhis Arrays by Ranges

The History array is summed up for the various user defined ranges. This summation represents the average number of penetrating particles per unit time for a specific range of elements (representing an SSCE, for example). The summation of the Arhis array represents the effective area for each range of elements.

7.5.12 Calculate PNP for Each Range

Given the spacecraft exposure time and the average number of penetrating particles per unit time for each range, the PNP for each range can be calculated. A Poisson model, as shown in equation 2, is used.

7.5.13 Write PNP and Effective Area Out

The PNP and effective area for each range of element IDs as well as the total PNP and effective area, are written out to both the screen and the summary file.

7.5.14 SUPERTAB Output File

A SUPERTAB Universal File containing the probability of penetration per surface area for each element in the model can be produced. This allows for penetration contour plots on the model to be made.

7.6 CONTOUR VERSION 2.0

CONTOUR produces a data base that can be used with user-supplied software to produce design contour plots of PNP versus shield and vessel wall thickness for given shield standoff, element ID range, and property ID. The code is a modified version of the BUMPER code. The main modification being the incorporation of the RESPONSE code as a subroutine. Figure 7.6-1 shows a simplified flow chart of the code. Those items that are different from the previous discussion of the BUMPER code will be discussed in more detail in the following sections. The code is well documented internally, and a complete listing is contained in appendix E.

7.6.1 Read in User Inputs

The user inputs are the same as those in the BUMPER and RESPONSE codes. Additionally, the minimum, maximum, and increment are input for the shield and vessel wall thickness. The code is limited to one range of element IDs and one property ID. If, during the analysis, elements in the specified range are encountered with the incorrect property ID, then the code skips the element. A warning message is written to the summary file indicating how many elements with the incorrect property ID were encountered.

7.6.2 Set Vessel Wall Thickness

The vessel wall thickness (Tb) is set using the user-defined minimum, maximum, and increment.

7.6.3 Set Shield Thickness

The shield thickness (Ts) is set using the user-defined minimum, maximum, and increment.

7.6.4 Ratio OK

The ratio of the shield thickness to the combined shield and vessel wall thickness is calculated. This ratio is limited to a range of 0.10 to 0.50. This is due to limitations in the various

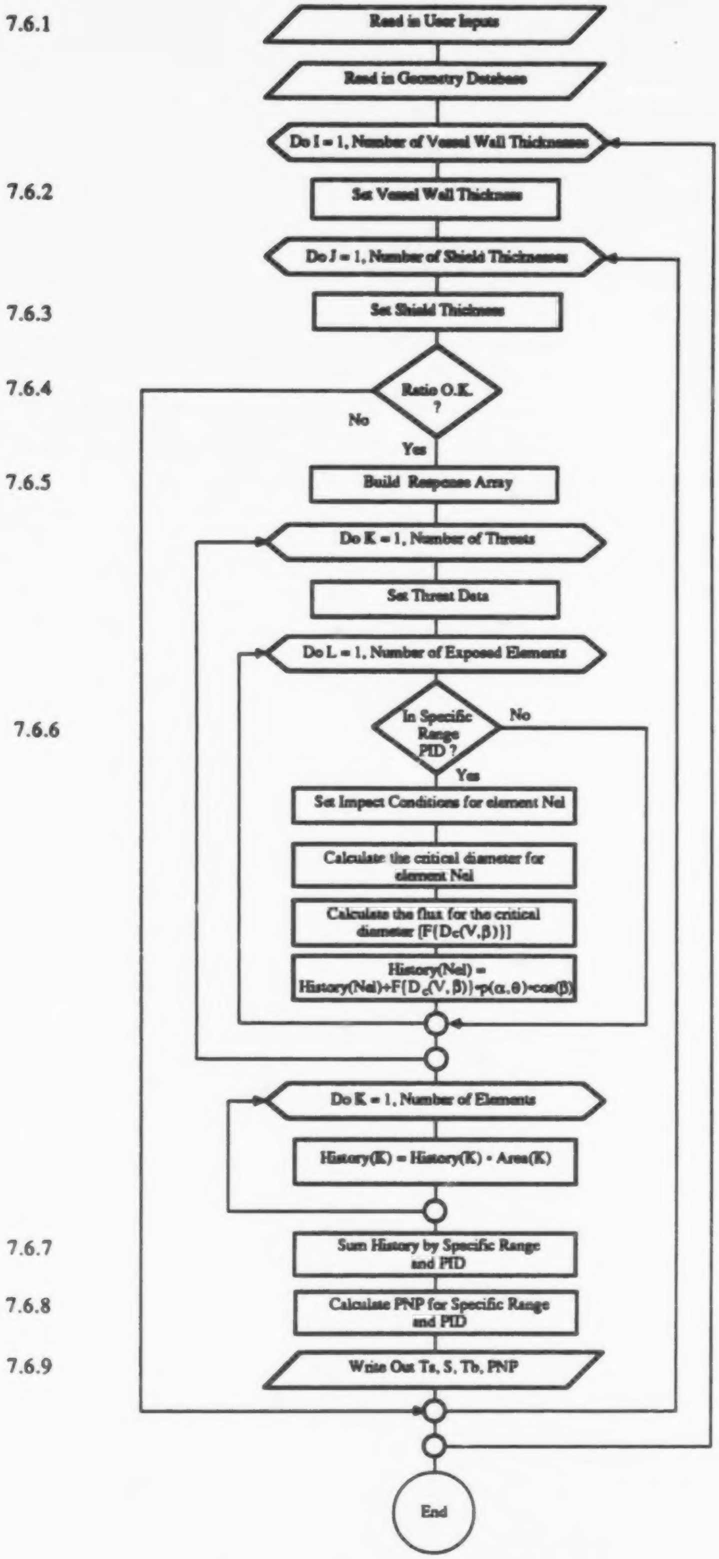

7.6.1
Read in User Inputs
Read in Geometry Database
Do I = 1, Number of Vessel Wall Thicknesses
7.6.2
Set Vessel Wall Thickness
Do J = 1, Number of Shield Thicknesses
7.6.3
Set Shield Thickness
7.6.4
Ratio O.K. ?
No
Yes
7.6.5
Build Response Array
Do K = 1, Number of Threats
Set Threat Data
Do L = 1, Number of Exposed Elements
7.6.6
In Specific Range PID ?
No
Yes
Set Impact Conditions for element Nel
Calculate the critical diameter for element Nel
Calculate the flux for the critical diameter {F(Dc(V,β))}
History(Nel) = History(Nel)+F{Dc(V,β))}•p(α,θ)•cos(β)
Do K = 1, Number of Elements
History(K) = History(K) • Area(K)
7.6.7
Sum History by Specific Range and PID
7.6.8
Calculate PNP for Specific Range and PID
7.6.9
Write Out Ts, S, Tb, PNP
End

Figure 7.6-1. CONTOUR Flow Chart.

penetration functions. If the ratio is outside of the range, then this combination is skipped and the next one is calculated.

7.6.5 Build Response Array

The response array (RTABLE) is built using the RESPONSE code as a subroutine in the CONTOUR code. The only modification to the RESPONSE code is the elimination of the input and output subroutines.

7.6.6 Inside Range

If the element is not within the specified element ID range or if the element is within the range but has a property ID other than the one specified, then the element is skipped and the next element evaluated.

7.6.7 Sum History by Specific Range and PID

For the specific range of element IDs, the History array is summed up. This represents the total average number of penetrating particles per unit time for the specified range.

7.6.8 Calculate the PNP for the Specific Range and PID

The PNP for the range is calculated using the user specified exposure time.

7.6.9 Write Out Ts, S, Tb, PNP

For each shield and vessel wall combination (within the specified bounds stated in sec. 7.6.4), the shield thickness (Ts), shield standoff (S), vessel wall thickness (Tb), and associated PNP is written out to the screen and the summary file.

REFERENCES

1 "Final Report-Space Station Integrated Wall Design and Penetration Damage Control," Contract NAS8-36426, The Boeing Company, July 1987.

APPENDIX A

SUPERTAB Universal File Format

Copyright 1983, Structural Dynamics Research Corporation

PRECEDING PAGE BLANK NOT FILMED

```
    -1
    2
SDRC__I-DEAS 2.5B: M

ME  2.54000E-02
    -1
    -1
    18
         6         0         0        15         1
REF
 -1.00000E+02  0.00000E+00 -1.00000E+02 -1.00000E+02 -5.00000E+00 -1.0
 -1.00000E+02 -5.00000E+00  0.00000E+00
    -1
    -1
    15
         1         6         6        11  4.36746E+02  7.45440E+00  6.
         2         6         6        11  4.36746E+02  6.43840E+00  5.
         .         .         .         .        .            .
         .         .         .         .        .            .
         .         .         .         .        .            .
         .         .         .         .        .            .
      2769         6         6        11  3.96919E+02  6.43840E+00  6.
      2770         6         6        11  3.96919E+02  6.91820E+00  6.
      2771         6         6        11  3.96919E+02  6.43840E+00  5.
    -1
    -1
    71
         1         2        91         1         1         7         3
       266        47        48
         2         2        91         1         1         7         3
        52       266       267
       .           .         .          .         .         .         .
       .           .         .          .         .         .         .
       .           .         .          .         .         .         .
       .           .         .          .         .         .         .
      5371         2        91         1         1         7         3
      2755      2770      2767
    -1
```

Figure A-1. SUPERTAB Example — Universal File Format

```
    -1
    73
SUPERTAB 8.0 - MS - 0729830004
         0           1          13           8
 0.100000E+01 0.000000E+00 0.000000E+00 0.000000E+00 0.100000E-34 0.10
 0.100000E-34 0.000000E+00 0.000000E+00 0.000000E+00 0.100000E-34 0.10
 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.00
 0.000000E+00
         0           5          13           8
 0.100000E+01 0.000000E+00 0.000000E+00 0.000000E+00 0.100000E-34 0.10
 0.100000E-34 0.000000E+00 0.000000E+00 0.000000E+00 0.100000E-34 0.10
 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.000000E+00 0.00
 0.000000E+00
         1           1           1          11
 0.206800E+12 0.290000E+00 0.782000E+03 0.100000E-34 0.361000E-05 0.10
 0.100000E-34 0.000000E+00 0.100000E-34 0.000000E+00 0.360000E+05
    -1
    -1
    74
SUPERTAB 8.0 - MS - 0729830004
         1          91           1           1           1
         2          91           2           1           1
         .           .           .           .           .
         .           .           .           .           .
         .           .           .           .           .
         .           .           .           .           .
      5370          91        5370           1           1
      5371          91        5371           1           1
    -1
```

Figure A-1. SUPERTAB Example – Universal File Format (Continued)

DATASET TYPE: 2
DESCRIPTION: PROGRAM HEADER
SUPERTAB, OUTPUT DISPLAY, ETC.
RECORD 1: FORMAT (20A1)
FIELD 1 — 20 CHARACTER PROGRAM VERSION IDENTIFIER.
RECORD 2: FORMAT(80A1)
FIELD 1 — 80 CHARACTER MODEL IDENTIFICATION.
RECORD 3: FORMAT(2A1,E13.5)
FIELD 1 — MODEL UNITS OF DATA
IN = INCHES
MM = MILLIMETERS
ME = METERS
UD = USER DEFINED SCALE
FIELD 2 — MODEL UNITS CONVERSION FACTOR
(MODEL UNITS PER INCH)

EXAMPLE:

```
  -1
    2
SDRC/SUPERTAB X.Y.Z
EXAMPLE OF PROGRAM HEADER DATASET
IN 1.00000E+00
  -1
```

DATASET TYPE: 15
DESCRIPTION: NODES
RECORD 1: FORMAT(4I10,3E13.5)
FIELD 1 — NODE TAG (NUMBER)
FIELD 2 — DEFINITION COORDINATE SYSTEM
FIELD 3 — DISPLACEMENT COORDINATE SYSTEM
FIELD 4 — COLOR
FIELDS 5-7 — 3-DIMENSIONAL COORDINATES OF NODE IN THE DEFINITION SYSTEM

RECORD 1 IS REPEATED FOR EACH NODE IN THE MODEL.

EXAMPLE:

```
    -1
    15
             1         0         0         8  0.00000E+00  0.00000E+00  0.00000E+00
             2         0         0         8  5.00000E-01  0.00000E+00 -5.00000E-02
             .
             .
             .
           100         0         0         8  1.20000E+01  1.20000E+01 -4.50000E+00
    -1
```

DATASET TYPE: 18
DESCRIPTION: COORDINATE SYSTEMS
RECORD 1: FORMAT(5I10)
FIELD 1 — COORDINATE SYSTEM TAG (NUMBER)
FIELD 2 — COORDINATE SYSTEM TYPE
FIELD 3 — REFERENCE COORDINATE SYSTEM TAG
FIELD 4 — COLOR
FIELD 5 — METHOD OF DEFINITION
RECORD 2: FORMAT(20A1)
FIELD 1 — 20 CHARACTER COORDINATE SYSTEM IDENTIFICAION FOR METHOD OF DEFINITION = 1
RECORD 3: FORMAT(6E13.5)
TOTAL OF 9 COORDINATE SYSTEM DEFINITION PARAMETERS.
FIELDS 1-3 — ORIGIN OF NEW SYSTEM SPECIFIED IN REFERENCE SYSTEM
FIELDS 4-6 — POINT ON +X AXIS OF THE NEW SYSTEM SPECIFIED IN REFERENCE SYSTEM
FIELDS 7-9 — POINT ON +XZ PLANE OF THE NEW SYSTEM SPECIFIED IN REFERENCE SYSTEM

REPEAT RECORDS 1 THRU 3 FOR ALL COORDINATE SYSTEMS.
EXAMPLE:

```
    -1
    18
         3         1         0         8         1
SYS1
 0.00000E+00 -1.00000E+00  7.00000E+00 -5.00000E+00  0.00000E+00 3.0000E+00
 6.00000E+00  4.50000E+00  3.00000E+00
    -1
```

DATASET TYPE: 56
DESCRIPTION: ANALYSIS DATA ON ELEMENTS
RECORD 1: FORMAT (80A1)
FIELD 1 — ID LINE 1
RECORD 2: FORMAT (80A1)
FIELD 1 — ID LINE 2
RECORD 3: FORMAT (80A1)
FIELD 1 — ID LINE 3
RECORD 4: FORMAT (80A1)
FIELD 1 — ID LINE 4
RECORD 5: FORMAT (80A1)
FIELD 1 — ID LINE 5
RECORD 6: FORMAT (6I10)

DATA DEFINITION PARAMETERS

FIELD 1 — MODEL TYPE
0: UNKNOWN
1: STRUCTURAL
2: HEAT TRANSFER
3: FLUID FLOW
FIELD 2 — ANALYSIS TYPE
0: UNKNOWN
1: STATIC
2: NORMAL MODE
3: COMPLEX EIGENVALUE
4: TRANSIENT
5: FREQUENCY RESPONSE
6: BUCKLING
FIELD 3 — DATA CHARACTERISTIC
0: UNKNOWN
1: SCALAR
2: 3 DOF GLOBAL TRANSLATION VECTOR
3: 6 DOF GLOBAL TRANSLATION & ROTATION VECTOR
4: SYMMETRIC GLOBAL TENSOR
5: GENERAL GLOBAL TENSOR
FIELD 4 — SPECIFIC DATA TYPE
0: UNKNOWN
1: UNKNOWN
2: STRESS
3: STRAIN
4: ELEMENTAL FORCE
5: TEMPERATURE
6: HEAT FLUX
7: STRAIN ENERGY
8: DISPLACEMENT
9: REACTION FORCE
10: KINETIC ENERGY
11: VELOCITY
12: ACCELERATION
FIELD 5 — DATA TYPE
2: REAL
5: COMPLEX
FIELD 6 — NUMBER OF DATA VALUES FOR EACH POSITION ON THE ELEMENT (NDV)

RECORDS 7 AND 8 ARE ANALYSIS TYPE SPECIFIC

GENERAL FORM

RECORD 7: FORMAT (8I10)
FIELD 1 — NUMBER OF INTEGER DATA VALUES
1 < OR = NINT <x OR = 10
FIELD 2 — NUMBER OF REAL DATA VALUES
1 < OR = NRVAL < OR = 12
FIELDS 3-N — TYPE SPECIFIC INTEGER PARAMETERS
RECORD 8: FORMAT (6E13.5)
FIELDS 1-N — TYPE SPECIFIC REAL PARAMETERS

FOR ANALYSIS TYPE = 0, UNKNOWN

RECORD 7:
FIELD 1 — 1
FIELD 2 — 1
FIELD 3 — ID NUMBER
RECORD 8:
FIELD 1 — 0.0

FOR ANALYSIS TYPE = 1, STATIC

RECORD 7:
FIELD 1 — 1
FIELD 2 — 1
FIELD 3 — LOAD CASE NUMBER
RECORD 8:
FIELD 4 — 0.0

FOR ANALYSIS TYPE = 2, NORMAL MODE

RECORD 7:
FIELD 1 — 2
FIELD 2 — 3
FIELD 3 — LOAD CASE NUMBER
FIELD 4 — MODE NUMBER
RECORD 8:
FIELD 1 — FREQUENCY (HERTZ)
FIELD 2 — MODAL MASS
FIELD 3 — MODAL DAMPING

FOR ANALYSIS TYPE = 3, COMPLEX EIGENVALUE

RECORD 7:
FIELD 1 — 2
FIELD 2 — 6
FIELD 3 — LOAD CASE NUMBER
FIELD 4 — MODE NUMBER
RECORD 8:
FIELD 1 — REAL PART EIGENVALUE
FIELD 2 — IMAGINARY PART EIGENVALUE
FIELD 3 — REAL PART OF MODAL A
FIELD 4 — IMAGINARY PART OF MODAL A
FIELD 5 — REAL PART OF MODAL B
FIELD 6 — IMAGINARY PART OF MODAL B

FOR ANALYSIS TYPE = 4, TRANSIENT

RECORD 7:
FIELD 1 — 2
FIELD 2 — 1
FIELD 3 — LOAD CASE NUMBER
FIELD 4 — TIME STEP NUMBER
RECORD 8:
FIELD 1 — TIME (SECONDS)

FOR ANALYSIS TYPE = 5, FREQUENCY RESPONSE

RECORD 7:
FIELD 1 — 2
FIELD 2 — 1
FIELD 3 — LOAD CASE NUMBER
FIELD 4 — FREQUENCY STEP NUMBER
RECORD 8:
FIELD 1 — FREQUENCY (HERTZ)

FOR ANALYSIS TYPE = 6, BUCKLING

RECORD 7:
FIELD 1 — 1
FIELD 2 — 1
FIELD 3 — LOAD CASE NUMBER
RECORD 8:
FIELD 1 — EIGENVALUE
RECORD 9: FORMAT (2I10)
FIELD 1 — ELEMENT NUMBER
FIELD 2 — NUMBER OF DATA VALUES FOR THIS ELEMENT (NVAL)
RECORD 10: FORMAT(6E13.5)
FIELDS 1-N — DATA ON ELEMENT (NVAL REAL OR COMPLEX VALUES)

RECORDS 9 AND 10 ARE REPEATED FOR ALL ELEMENTS.

NOTES:

1. D LINES MAY NOT BE BLANK. IF NO INFORMATION IS "NONE" MUST APPEAR IN COLUMNS 1-4.
2. FOR COMPLEX DATA THERE WILL BE 2*NVAL DATA ITEMS. THE ORDER IS REAL PART FOR VALUE 1, IMAGINARY PART FOR VALUE 1, REAL PART FOR VALUE 2, IMAGINARY PART FOR VALUE 2, ETC.
3. THE ORDER OF VALUES FOR VARIOUS DATA CHARACTERISTICS IS:
   3 DOF GLOBAL VECTOR: X, Y, Z
   6 DOF GLOBAL VECTOR: X, Y, Z, RX, RY, RZ
   SYMMETRIC GLOBAL TENSOR: SXX, SXY, SYY, SXZ, SYZ, SZZ
   GENERAL GLOBAL TENSOR: SXX, SYX, SZX, SXY, SYY, SZY, SXZ, SYZ, SZZ
4. ID LINE 1 ALWAYS APPEARS ON PLOTS IN OUTPUT DISPLAY.
5. IF SPECIFIC DATA TYPE IS "UNKNOWN," ID LINE 2 IS DISPLAYED AS DATA TYPE IN OUTPUT DISPLAY.
6. TYPICAL FORTRAN I/O STATEMENTS FOR THE DATA SECTIONS ARE:

```
     READ (LUN, 1000) NUM, NVAL
     WRITE
1000 FORMAT (2I10)
     READ (LUN, 1010) (VAL(I),I = 1,NVAL)
     WRITE
1010 FORMAT (6E13.5)
```

   WHERE: NUM IS ELEMENT NUMBER
   NVAL IS NUMBER OF REAL OR COMPLEX DATA VALUES FOR THIS ELEMENT (MAX = 90)
   VAL IS REAL OR COMPLEX DATA ARRAY
7. DATA CHARACTERISTIC VALUES IMPLY THE FOLLOWING VALUES OF NDV:
   SCALAR: 1
   3 DOF GLOBAL VECTOR: 3
   6 DOF GLOBAL VECTOR: 6
   SYMMETRIC GLOBAL TENSOR: 6
   GENERAL GLOBAL TENSOR: 9
8. DATA ON 2D TYPE ELEMENTS MAY HAVE MULTIPLE VALUES THROUGH THE ELEMENT THICKNESS. IN THESE CASES, NVAL = NDVNPOS WHERE NPOS IS NUMBER OF POSITIONS THROUGH ELEMENT. NPOS IS ALWAYS 1 FOR SOLIDS. THE ORDER OF THE DATA IS NDV VALUES FOR POSITION 1, NDV VALUES FOR POSITION 2, ETC. THE ORDER OF THE NODES DEFINES AN OUTWARD NORMAL WHICH SPECIFIES THE ORDER FROM POSITION 1 TO NPOS.

9. ANY RECORD WITH ALL 0.0'S DATA ENTRIES NEED NOT (BUT MAY) APPEAR.
10. A DIRECT RESULT OF 9 IS THAT IF NO RECORD 9 & 10 APPEARS, ALL DATA FOR THE DATA SET IS 0.0.
11. WHEN NEW ANALYSIS TYPES ARE ADDED, RECORD 7 FIELDS 1 AND 2 ARE ALWAYS < OR = 1 WITH DUMMY INTERGER AND REAL ZERO DATA IF DATA IS NOT REQUIRED. IF COMPLEX DATA IS NEEDED, IT IS TREATED AS TWO REAL NUMBERS, REAL PART FOLLOWED BY IMAGINARY POINT.
12. DATALOADERS USE THE FOLLOWING ID LINE CONVENTION:
    1. (80A1) MODEL IDENTIFICATION
    2. (80A1) RUN IDENTIFICATION
    3. (80A1) RUN DATE/TIME
    4. (80A1) LOAD CASE NAME

    FOR STATIC:

    5. (17H LOAD CASE NUMBER;, I10)

    FOR NORMAL MODE:

    5. (10H MODE SAME, I10, 10H FREQUENCY, E13.5)
13. MAXIMUM VALUE FOR NDV IS 9.
    MAXIMUM VALUE FOR NVAL IS 90.
14. TYPICAL FORTRAN I/O STATEMENTS FOR PROCESSING RECORDS 7 AND 8.

```
     READ (LUN, 1000) NINT, NRVAL, (IPAR(I),I = 1,NINT)
1000 FORMAT (8I10)
     READ (LUN,1010) (NRVAL(I),I = 1,NRVAL)
1010 FORMAT (6E13.5)
```

DATASET TYPE: 71
DESCRIPTION: CONNECTIVITY
RECORD 1: FORMAT(7I10)
FIELD 1 — ELEMENT TAG (NUMBER)
FIELD 2 — CONNECTIVITY NUMBER
FIELD 3 — TYPE REFERENCE (NUMBER)
FIELD 4 — PROPERTY REFERENCE (NUMBER)
FIELD 5 — MATERIAL REFERENCE (NUMBER)
FIELD 6 — COLOR
FIELD 7 — NUMBER OF NODES ON ELEMENT
RECORD 2: FORMAT (8I10)
FIELDS 1-N — NODE NUMBERS OF NODES DEFINING ELEMENT

THE ENTIRE SET OF RECORDS IS REPEATED FOR EACH ELEMENT OF THE MODEL.

EXAMPLE:

```
    -1
    71
         1        19         1         1         1         8         8
        11        12        13        16        21        20        19        15
         2        12         2         2         1         8        16
        31        32        33        34        35        36        37        38
        39        40        41        42        43        44        45        46
         .
         .
         .
       124        19         1         1         1         8         8
         9        10        11        15        19        18        17        14
    -1
```

DATASET TYPE: 73
DESCRIPTION: ELEMENT PROPERTY VALUE ENTRIES
RECORD 1: FORMAT(40A2)
FIELD 1 — PARENT FINITE ELEMENT SET SERIAL NUMBER
RECORD 2: FORMAT (4I10)
FIELD 1 — ENTRY TYPE
0 — PHYSICAL
1 — ISOTROPIC MATERIAL
2 — ORTHOTROPIC MATERIAL
3 — ANISOTROPIC MATERIAL
FIELD 2 — ENTRY ID
FIELD 3 — PHYSICAL OR MATERIAL DESCRIPTOR ID
FIELD 4 — # OF PROPERTIES IN VALUE ENTRY
RECORD 3: FORMAT (6E13.6)
FIELD 1<n<6 — VALUES AS NEEDED
RECORD 4: FORMAT(I10,35A2)
FIELD 1 — SYMBOL COUNT
FIRST RECORD OF VALUE — TOTAL SYMBOLS IN VALUE
EACH SUCCESSIVE RECORD — TOTAL FOR THAT RECORD

ONE RECORD TYPE 1 APPEARS PER DATASET. ONE RECORD TYPE 2 APPEARS FOR EACH VALUE ENTRY IN THE DATASET. AS MANY RECORD TYPE 3 AND RECORD TYPE 4 APPEAR PER RECORD TYPE 2 AS IS NECESSARY TO TRANSFER THE PROPERTY VALUES IN EACH ENTRY.

FINITE ELEMENT PHYSICAL PROPERTIES

PROPERTY DESCRIPTOR	ELEMENT FAMILY AND INDEXES	
1	NULL PROPERTY TABLE NUMBER OF PROPERTIES = 0	NUMBER OF VALUES = 0
2	LINEAR BEAM NUMBER OF PROPERTIES = 37 Property Indices 2,3,4,7,26,5,6,8,9,10,11,12,19,13,20,14,15,16,17, 18,51,58,94,59,95,60,96,25,27,83,84,85,86,87,91,92,93	NUMBER OF VALUES = 57
3	NODE TO NODE SPRING NUMBER OF PROPERTIES = 8 Property Indices 22,21,63,64,53,54,55,90	NUMBER OF VALUES = 15
4	SOLID NUMBER OF PROPERTIES = 3 Property Indices 49,88,89	NUMBER OF VALUES = 5
5	ROD NUMBER OF PROPERTIES = 4 Property Indices 2,7,24,25	NUMBER OF VALUES = 4
6	TAPERED BEAM NUMBER OF PROPERTIES = 65 Property Indices 28,30,32,97,99,38,34,65,107,67,109,68,111,69,113 70,115,75,40,79,77,101,103,105,44 10,11,12,19,13,20,14,15,16,17,36,37,42,43,81 29,31,33,98,100,39,35,66,108,71,110,72,112,73,114,74 116,76,41,80,78,102,104,106,45	NUMBER OF VALUES = 93
7	CURVED BEAM NUMBER OF PROPERTIES = 10 Property Indices 2,3,4,7,5,6,18,25,8,9	NUMBER OF VALUES = 11
8	PIPE NUMBER OF PROPERTIES = 5 Property Indices 191,192,193,194,25	NUMBER OF VALUES = 5
9	PHYSICAL PROPERTIES FOR DATA FORMATTER NUMBER OF PROPERTIES = 49 Property Indices 201,202,203,204,205,206,207,209,209,210,211,212,213 214,215,216,217,218,219,220,221,222,223,224,225,226 227,228,229,230,231,232,233,234,235,236,237,238,239 240,241,242,243,244,245,246,247,248,249	NUMBER OF VALUES = 48

10	PLANE STRESS NUMBER OF PROPERTIES = 2 Property Indices 1,49	NUMBER OF VALUES = 7

PROPERTY DESCRIPTOR	ELEMENT FAMILY AND PROPERTY NUMBERS	
11	PLANE STRAIN NUMBER OF PROPERTIES = 2 Property Indices 1,49	NUMBER OF VALUES = 7
12	AXISYMMETRIC NUMBER OF PROPERTIES = 1 Property Indices 49	NUMBER OF VALUES = 3
13	THIN SHELL NUMBER OF PROPERTIES = 7 Property Indices 1,46,47,48,49,52,82	NUMBER OF VALUES = 11
14	FLAT PLATE NUMBER OF PROPERTIES = 3 Property Indices 1,46,49	NUMBER OF VALUES = 6
15	LUMPED MASS NUMBER OF PROPERTIES = 6 Property Indices 23,56,61,62,57,54	NUMBER OF VALUES = 13
16	THICK SHELL NUMBER OF PROPERTIES = 1 Property Indices 49	NUMBER OF VALUES = 3
17	NODE TO GROUND SPRING NUMBER OF PROPERTIES = 5 Property Indices 22,21,53,54,90	NUMBER OF VALUES = 12
18	NODE TO NODE DAMPER NUMBER OF PROPERTIES = 3 Property Indices 53,54,55	NUMBER OF VALUES = 3
19	NODE TO GROUND DAMPER NUMBER OF PROPERTIES = 2 Property Indices 53,54	NUMBER OF VALUES = 2

FINITE ELEMENT PHYSICAL PROPERTIES

INDEX	MENU PICK	DESCRIPTION
1	TK	THICKNESS (CAN INPUT UP TO 4 VALUES ANY 0.0 VALUE DEFAULTS TO PREVIOUS VALUE) DEFAULT VALUES 1.0,0.0,0.0,0.0
2	AR	AREA — CROSS SECTION AREA
3	IYY	MOMENT OF INERTIA ABOUT Y AXES
4	IZZ	MOMENT OF INERTIA ABOUT Z AXES
5	SRY	SHEAR AREA RATIO IN THE Y DIRECTION (SUPERB)
6	SRZ	SHEAR AREA RATIO IN THE Z DIRECTION (SUPERB)
7	TC	TORSIONAL CONSTANT
8	ECY	ECCENTRICITY Y — Y DISTANCE FROM SHEAR CENTER TO CENTROID (SUPERB)
9	ECZ	ECCENTRICITY Z — Z DISTANCE FROM SHEAR CENTER TO CENTROID (SUPERB)
10	OC	ORIENTATION CODE
11	OS	ORIENTATION SPECIFICATION
12	RFE	FORE END RELEASE FUNCTION (SUPERB.NASTRAN)
13	RAE	AFT END RELEASE FUNCTION (SUPERB.NASTRAN)
14	OFCF	OFFSET CODE FORE END (SUPERB)
15	OFCA	OFFSET CODE AFT END (SUPERB)
16	OFVF	OFFSET VECTOR FORE END
17	OFVA	OFFSET VECTOR AFT END
18	SRC	STRESS RECOVERY C (2 VALUES) (SUPERB,NASTRAN)

19	SFF	SPRING FUNCTION FORE END (6 VALUES) (SUPERB)
20	SFA	SPRING FUNCTION AFT END (6 VALUES) (SUPERB)
21	KIND	SPRING COEFFICIENT CODE
22	K	SPRING COEFFICIENT (8 VALUES)
23	M	MASS
24	CTS	COEFFICIENT FOR TORSIONAL STRESS
25	NSML	NONSTRUCTURAL MASS PER UNIT LENGTH (NASTRAN)
26	IXY	PRODUCT OF INERTIA (NASTRAN)
27	SAL	SURFACE AREA PER UNIT LENGTH (SUPERB)
28	ARF	AREA FORE END
29	ARA	AREA AFT END
30	IYYF	MOMENT OF INERTIA ABOUT Y AXES FORE END
31	IYYA	MOMENT OF INERTIA ABOUT Y AXES AFT END
32	IZZF	MOMENT OF INERTIA ABOUT Z AXES FORE END
33	IZZA	MOMENT OF INERTIA ABOUT Z AXES AFT END
34	TCF	TORSIONAL CONSTANT FORE END
35	TCA	TORSIONAL CONSTANT AFT END
36	SSF1	SHEAR STIFFNESS FACTOR XY PLANE
37	SSF2	SHEAR STIFFNESS FACTOR XZ PLANE
38	IXYF	PRODUCT OF INERTIA FORE END
39	IXYA	PRODUCT OF INERTIA AFT END
40	WCF	WARPING COEFFICIENT FORE END
41	WCA	WARPING COEFFICIENT AFT END
42	SRT1	SHEAR RELIEF DUE TO TAPER XY PLANE
43	SRT2	SHEAR RELIEF DUE TO TAPER XZ PLANE
44	NAF	COORDINATES OF NEUTRAL AXIS FORE END
45	NAA	COORDINATES OF NEUTRAL AXIS AFT END
46	BSP	BENDING STIFFNESS PARAMETER (NASTRAN)
47	TSMT	TRANSVERSE SHEAR THICKNESS DIVIDED BY MEMBRANE THICKNESS (NASTRAN)
48	NSMA	NONSTRUCTURAL MASS PER UNIT AREA (NASTRAN)
49	MOV	MATERIAL ORIENTATION VECTOR, REAL, 3 VALUES
51	ERTC	EFFECTIVE RADIUS IN TORSION C (SUPERB,SAGS)
52	Z12	FIBER DISTANCES FOR STRESS COMPUTATION, 2 VALUES (NASTRAN)
53	GE	DAMPING COEFFICIENT (NASTRAN)
54	C1	COMPONENT NUMBER 1 (NASTRAN)
55	C2	COMPONENT NUMBER 2 (NASTRAN)
56	MICS	MASS INERTIA COORDINATE SYSTEM (SUPERB)
57	MIM	MASS INERTIA MATRIX (SUPERB)
58	SRD	STRESS RECOVERY D (2 VALUES) (NASTRAN)
59	SRE	STRESS RECOVERY E (2 VALUES) (NASTRAN)
60	SRF	STRESS RECOVERY F (2 VALUES) (NASTRAN)
61	MOC	MASS OFFSET CODE (SUPERB)
62	MO	MASS OFFSET (3 VALUES) SUPERB
63	SOC	SPRING ORIENTATION CODE (SUPERB)
64	SOS	SPRING ORIENTATION SPECIFICATON (3 VALUES) SUPERB
65	NSMF	NONSTRUCTURAL MASS PER UNIT LENGTH FORE END (NASTRAN)
66	NSMA	NONSTRUCTURAL MASS PER UNIT LENGTH AFT END (NASTRAN)
67	SRCF	STRESS RECOVERY C FORE END (2 VALUES) (NASTRAN)
68	SRDF	STRESS RECOVERY D FORE END (2 VALUES) (NASTRAN)
69	SREF	STRESS RECOVERY E FORE END (2 VALUES) (NASTRAN)
70	SRFF	STRESS RECOVERY F FORE END (2 VALUES) (NASTRAN)
71	SRCA	STRESS RECOVERY C AFT END (2 VALUES) (NASTRAN)
72	SRDA	STRESS RECOVERY D AFT END (2 VALUES) (NASTRAN)

73	SREA	STRESS RECOVERY E AFT END (2 VALUES) (NASTRAN)
74	SRFA	STRESS RECOVERY F AFT END (2 VALUES) (NASTRAN)
75	NSIF	NONSTRUCTURAL MASS MOMENT OF INERTIA FORE END (NASTRAN)
76	NSIA	NONSTRUCTURAL MASS MOMENT OF INERTIA AFT END (NASTRAN)
77	NCGF	NONSTRUCTURAL MASS CENTER OF GRAVITY FORE END (NASTRAN)
78	NCGA	NONSTRUCTURAL MASS CENTER OF GRAVITY AFT END (NASTRAN)
79	WSAF	WARPING VARIABLE POINT FORE END (NASTRAN)
80	WSBA	WARPING VARIABLE POINT AFT END (NASTRAN)
81	SOO	STRESS OUTPUT OPTION (ALPHANUMERIC) (NASTRAN)
82	EFS	ELASTIC FOUNDATION STIFFNESS (ANSYS)
83	TKZ	BEAM THICKNESS Z DIRECTION (ANSYS)
84	THY	BEAM THICKNESS Y DIRECTION (ANSYS)
85	IS	INITIAL STRAIN (ANSYS)
86	SDCZ	SHEAR DEFLECTION CONSTANT Z DIRECTION (ANSYS)
87	SDCY	SHEAR DEFLECTION CONSTANT Y DIRECTION (ANSYS)
88	IN	INTEGRATION NETWORK (NASTRAN)
89	LOST	LOCATION FOR STRESS OUTPUT (NASTRAN)
90	SSC	SPRING STRESS COEFFICIENT (ANSYS)
91	WC	WARPING COEFFICIENT (SAGS)
92	FIX	DEGREE OF FIXITY (SAGS)
93	CSC	COMBINED STRESS CODE (SAGS)
94	ERTD	EFFECTIVE RADIUS IN TORSION D (SAGS)
95	ERTF	EFFECTIVE RADIUS IN TORSION E (SAGS)
96	ERTF	EFFECTIVE RADIUS IN TORSION F (SAGS)
97	SRYF	SHEAR AREA RATION Y FORE END (SAGS)
98	SRYA	SHEAR AREA RATION Y AFT END (SAGS)
99	SRZF	SHEAR AREA RATION Z FORE END (SAGS)
100	SRZA	SHEAR AREA RATION Z AFT END (SAGS)
101	FIXF	DEGREE OF FIXITY FORE END (SAGS)
102	FISA	DEGREE OF FIXITY AFT END (SAGS)
103	ECYF	ECCENTRICITY Y FORE END (SAGS)
104	ECYA	ECCENTRICITY Y AFT END (SAGS)
105	ECZF	ECCENTRICITY Z FORE END (SAGS)
106	ECZA	ECCENTRICITY Z AFT END (SAGS)
107	CSCF	COMBINED STRESS CODE FORE END (SAGS)
108	CSCA	COMBINED STRESS CODE AFT END (SAGS)
109	ERCF	EFFECTIVE RADIUS IN TORSION C FORE END (SAGS)
110	ERCA	EFFECTIVE RADIUS IN TORSION C AFT END (SAGS)
111	ERDF	EFFECTIVE RADIUS IN TORSION D FORE END (SAGS)
112	ERDA	EFFECTIVE RADIUS IN TORSION D AFT END (SAGS)
113	EREF	EFFECTIVE RADIUS IN TORSION E FORE END (SAGS)
114	EREA	EFFECTIVE RADIUS IN TORSION E AFT END (SAGS)
115	ERFF	EFFECTIVE RADIUS IN TORSION F FORE END (SAGS)
116	ERFA	EFFECTIVE RADIUS IN TORSION F AFT END (SAGS)
191	OD	OUTER DIAMETER — PIPE PROPERTY
192	WT	WALL THICKNESS — PIPE PROPERTY
193	SIFF	STRESS INTENSITY FACTOR FORE END — PIPE PROPERTY
194	SIFA	STRESS INTENSITY FACTOR AFT END — PIPE PROPERTY
201	P1	PHYSICAL PROPERTY 1 FOR DATA FORMATTER
202	P2	PHYSICAL PROPERTY 2 FOR DATA FORMATTER
203	P3	PHYSICAL PROPERTY 3 FOR DATA FORMATTER
204	P4	PHYSICAL PROPERTY 4 FOR DATA FORMATTER
205	P5	PHYSICAL PROPERTY 5 FOR DATA FORMATTER
206	P6	PHYSICAL PROPERTY 6 FOR DATA FORMATTER

207	P7	PHYSICAL PROPERTY 7 FOR DATA FORMATTER
208	P8	PHYSICAL PROPERTY 8 FOR DATA FORMATTER
209	P9	PHYSICAL PROPERTY 9 FOR DATA FORMATTER
210	P10	PHYSICAL PROPERTY 10 FOR DATA FORMATTER
211	P11	PHYSICAL PROPERTY 11 FOR DATA FORMATTER
212	P12	PHYSICAL PROPERTY 12 FOR DATA FORMATTER
213	P13	PHYSICAL PROPERTY 13 FOR DATA FORMATTER
214	P14	PHYSICAL PROPERTY 14 FOR DATA FORMATTER
215	P15	PHYSICAL PROPERTY 15 FOR DATA FORMATTER
216	P16	PHYSICAL PROPERTY 16 FOR DATA FORMATTER
217	P17	PHYSICAL PROPERTY 17 FOR DATA FORMATTER
218	P18	PHYSICAL PROPERTY 18 FOR DATA FORMATTER
219	P19	PHYSICAL PROPERTY 19 FOR DATA FORMATTER
220	P20	PHYSICAL PROPERTY 20 FOR DATA FORMATTER
221	P21	PHYSICAL PROPERTY 21 FOR DATA FORMATTER
222	P22	PHYSICAL PROPERTY 22 FOR DATA FORMATTER
223	P23	PHYSICAL PROPERTY 23 FOR DATA FORMATTER
224	P24	PHYSICAL PROPERTY 24 FOR DATA FORMATTER
225	P25	PHYSICAL PROPERTY 25 FOR DATA FORMATTER
226	P26	PHYSICAL PROPERTY 26 FOR DATA FORMATTER
227	P27	PHYSICAL PROPERTY 27 FOR DATA FORMATTER
228	P28	PHYSICAL PROPERTY 28 FOR DATA FORMATTER
229	P29	PHYSICAL PROPERTY 29 FOR DATA FORMATTER
230	P30	PHYSICAL PROPERTY 30 FOR DATA FORMATTER
231	P31	PHYSICAL PROPERTY 31 FOR DATA FORMATTER
232	P32	PHYSICAL PROPERTY 32 FOR DATA FORMATTER
232	P33	PHYSICAL PROPERTY 33 FOR DATA FORMATTER
234	P34	PHYSICAL PROPERTY 34 FOR DATA FORMATTER
235	P35	PHYSICAL PROPERTY 35 FOR DATA FORMATTER
236	P36	PHYSICAL PROPERTY 36 FOR DATA FORMATTER
237	P37	PHYSICAL PROPERTY 37 FOR DATA FORMATTER
238	P38	PHYSICAL PROPERTY 38 FOR DATA FORMATTER
239	P39	PHYSICAL PROPERTY 39 FOR DATA FORMATTER
240	P40	PHYSICAL PROPERTY 40 FOR DATA FORMATTER
241	P41	PHYSICAL PROPERTY 41 FOR DATA FORMATTER
242	P42	PHYSICAL PROPERTY 42 FOR DATA FORMATTER
243	P43	PHYSICAL PROPERTY 43 FOR DATA FORMATTER
244	P44	PHYSICAL PROPERTY 44 FOR DATA FORMATTER
245	P45	PHYSICAL PROPERTY 45 FOR DATA FORMATTER
246	P46	PHYSICAL PROPERTY 46 FOR DATA FORMATTER
247	P47	PHYSICAL PROPERTY 47 FOR DATA FORMATTER
248	P48	PHYSICAL PROPERTY 48 FOR DATA FORMATTER
249	P49	PHYSICAL PROPERTY 49 FOR DATA FORMATTER

FINITE ELEMENT MATERIAL PROPERTIES

PROPERTY DESCRIPTOR	ELEMENT FAMILY AND INDEXES
1	ISOTROPIC NUMBER OF PROPERTIES = 11 NUMBER OF VALUES = 11 Property Indices 501,502,503,504,505,506,507,508,509,510,617
2	ISOTROPIC (NULL) NUMBER OF PROPERTIES = 0 NUMBER OF VALUES = 0
3	ORTHOTROPIC NUMBER OF PROPERTIES = 19 NUMBER OF VALUES = 19 Property Indices 503,507,508,601,602,603,604,605,606,608,609,610,611, 612,613,614,615,616,617

4 ANISOTROPIC
NUMBER OF PROPERTIES = 12 NUMBER OF VALUES = 37
Property Indices 503,507,508,509,701,702,703,704,705,706,708,709

5 ORTHOTROPIC (NULL)
NUMBER OF PROPERTIES = 0 NUMBER OF VALUES = 0

6 ANISOTROPIC (NULL)
NUMBER OF PROPERTIES = 0 NUMBER OF VALUES = 0

7 ISOTROPIC (MATERIAL FOR DATA FORMATTER ELEMENTS)
NUMBER OF PROPERTIES = 49 NUMBER OF VALUES = 49
Property Indices 301,302,303,304,305,306,307,308,309,310,311,312,313
314,315,316,317,318,319,320,321,322,323,324,325,326
327,328,329,330,331,332,333,334,335,336,337,338,339
340,341,342,343,344,345,346,347,348,349

FINITE ELEMENT MATERIAL PROPERTIES

INDEX	MENU PICK	DESCRIPTION
501	E	MODULUS OF ELASTICITY
502	NU	POISSON RATIO
503	DEN	MASS DENSITY
504	G	SHEAR MODULUS
505	A	COEFFICIENT OF THERMAL EXPANSION
506	K	THERMAL CONDUCTIVITY
507	TREF	THERMAL EXPANSION REFERENCE TEMPERATURE (NASTRAN)
508	GE	STRUCTURAL ELEMENT DAMPING COEFFICIENT (NASTRAN)
509	CP	THERMAL CAPACITY PER UNIT VOLUME (NASTRAN)
510	YS	YIELD STRESS (DEFAULT = 36000.) (SAGS)
601	EX	MODULUS OF ELASTICITY X DIRECTION
602	EY	MODULUS OF ELASTICITY Y DIRECTION
603	EZ	MODULUS OF ELASTICITY Z DIRECTION
604	NUXY	POISSON RATIO XY PLANE
605	NUYZ	POISSON RATIO YZ PLANE
606	NUXZ	POISSON RATIO XZ PLANE
608	GXY	SHEAR MODULUS XY PLANE
609	GYZ	SHEAR MODULUS YZ PLANE
610	GXZ	SHEAR MODULUS XZ PLANE
611	AX	COEFFICIENT OF THERMAL EXPANSION X DIRECTION
612	AY	COEFFICIENT OF THERMAL EXPANSION Y DIRECTION
613	AZ	COEFFICIENT OF THERMAL EXPANSION Z DIRECTION
614	KX	THERMAL CONDUCTIVITY X DIRECTION
615	KY	THERMAL CONDUCTIVITY Y DIRECTION
616	KZ	THERMAL CONDUCTIVITY Z DIRECTION
617	Q	HEAT GENERATION RATE (SUPERB)
701	RW1	ROW 1 MATERIAL PROPERTY MATRIX (6 VALUES)
702	RW2	ROW 2 MATERIAL PROPERTY MATRIX (5 VALUES)
703	RW3	ROW 3 MATERIAL PROPERTY MATRIX (4 VALUES)
704	RW4	ROW 4 MATERIAL PROPERTY MATRIX (3 VALUES)
705	RW5	ROW 5 MATERIAL PROPERTY MATRIX (2 VALUES)
706	RW6	ROW 6 MATERIAL PROPERTY MATRIX (1 VALUE)
708	TEV	THERMAL EXPANSION VECTOR (6 VALUES)
709	KKM	THERMAL CONDUCTIVITY MATRIX (6 VALUES)
301	M1	MATERIAL PROPERTY 1 FOR DATA FORMATTER
302	M2	MATERIAL PROPERTY 2 FOR DATA FORMATTER
303	M3	MATERIAL PROPERTY 3 FOR DATA FORMATTER
304	M4	MATERIAL PROPERTY 4 FOR DATA FORMATTER

305	M5	MATERIAL PROPERTY 5 FOR DATA FORMATTER
306	M6	MATERIAL PROPERTY 6 FOR DATA FORMATTER
307	M7	MATERIAL PROPERTY 7 FOR DATA FORMATTER
308	M8	MATERIAL PROPERTY 8 FOR DATA FORMATTER
309	M9	MATERIAL PROPERTY 9 FOR DATA FORMATTER
310	M10	MATERIAL PROPERTY 10 FOR DATA FORMATTER
311	M11	MATERIAL PROPERTY 11 FOR DATA FORMATTER
312	M12	MATERIAL PROPERTY 12 FOR DATA FORMATTER
313	M13	MATERIAL PROPERTY 13 FOR DATA FORMATTER
314	M14	MATERIAL PROPERTY 14 FOR DATA FORMATTER
315	M15	MATERIAL PROPERTY 15 FOR DATA FORMATTER
316	M16	MATERIAL PROPERTY 16 FOR DATA FORMATTER
317	M17	MATERIAL PROPERTY 17 FOR DATA FORMATTER
318	M18	MATERIAL PROPERTY 18 FOR DATA FORMATTER
319	M19	MATERIAL PROPERTY 19 FOR DATA FORMATTER
320	M20	MATERIAL PROPERTY 20 FOR DATA FORMATTER
321	M21	MATERIAL PROPERTY 21 FOR DATA FORMATTER
322	M22	MATERIAL PROPERTY 22 FOR DATA FORMATTER
323	M23	MATERIAL PROPERTY 23 FOR DATA FORMATTER
324	M24	MATERIAL PROPERTY 24 FOR DATA FORMATTER
325	M25	MATERIAL PROPERTY 25 FOR DATA FORMATTER
326	M26	MATERIAL PROPERTY 26 FOR DATA FORMATTER
327	M27	MATERIAL PROPERTY 27 FOR DATA FORMATTER
328	M28	MATERIAL PROPERTY 28 FOR DATA FORMATTER
329	M29	MATERIAL PROPERTY 29 FOR DATA FORMATTER
330	M30	MATERIAL PROPERTY 30 FOR DATA FORMATTER
331	M31	MATERIAL PROPERTY 31 FOR DATA FORMATTER
332	M32	MATERIAL PROPERTY 32 FOR DATA FORMATTER
333	M33	MATERIAL PROPERTY 33 FOR DATA FORMATTER
334	M34	MATERIAL PROPERTY 34 FOR DATA FORMATTER
335	M35	MATERIAL PROPERTY 35 FOR DATA FORMATTER
336	M36	MATERIAL PROPERTY 36 FOR DATA FORMATTER
337	M37	MATERIAL PROPERTY 37 FOR DATA FORMATTER
338	M38	MATERIAL PROPERTY 38 FOR DATA FORMATTER
339	M39	MATERIAL PROPERTY 39 FOR DATA FORMATTER
340	M40	MATERIAL PROPERTY 40 FOR DATA FORMATTER
341	M41	MATERIAL PROPERTY 41 FOR DATA FORMATTER
342	M42	MATERIAL PROPERTY 42 FOR DATA FORMATTER
343	M43	MATERIAL PROPERTY 43 FOR DATA FORMATTER
344	M44	MATERIAL PROPERTY 44 FOR DATA FORMATTER
345	M45	MATERIAL PROPERTY 45 FOR DATA FORMATTER
346	M46	MATERIAL PROPERTY 46 FOR DATA FORMATTER
347	M47	MATERIAL PROPERTY 47 FOR DATA FORMATTER
348	M48	MATERIAL PROPERTY 48 FOR DATA FORMATTER
349	M49	MATERIAL PROPERTY 49 FOR DATA FORMATTER

DATASET TYPE: 74
DESCRIPTION: ELEMENT
RECORD 1: FORMAT(40A2)
FIELD 1 — PARENT FINITE ELEMENT SET SERIAL NUMBER
RECORD 2-N: FORMAT(5I10)
FIELD 1 — FINITE ELEMENT LABEL
FIELD 2 — FINITE ELEMENT ANCHOR TYPE ID
FIELD 3 — FINITE ELEMENT TOPOLOGY LABEL (IF ZERO, THEN ASSUMED TO BE SAME AS FINITE ELEMENT LABEL)
FIELD 4 — FINITE ELEMENT PHYSICAL VALUE ID
FIELD 5 — FINITE ELEMENT MATERIAL VALUE ID

FINITE ELEMENTS

ELEMENT	ANCHOR		PROPERTY DESCRIPTOR TYPES			
DESCRIPTION*	TYPE	CONNECTIVITY #	PHYSICAL	ISO	ORTHO	ANISO
PSS,LT	42	2	10	1	3	4
PSS,PT	42	3	10	1	3	4
PSS,CT	43	3	10	1	3	0
PSS,LQ	44	5	10	1	3	4
PSS,PQ	45	6	10	1	3	4
PSS,CQ	40	7	10	1	3	0
PST,LT	51	2	11	1	3	0
PST,PT	52	3	11	1	3	0
PST,CT	53	4	11	1	3	0
PST,LQ	54	5	11	1	3	0
PST,PQ	55	6	11	1	3	0
PST,CQ	56	7	11	1	3	0
PLT,LT	61	2	14	1	3	4
PLT,PT	62	3	14	1	3	4
PLT,CT	63	1	14	1	3	4
PLT,LQ	64	5	14	1	3	4
PLT,PQ	65	6	14	1	3	4
PLT,CQ	66	7	14	1	3	0
MEM,LQ	71	5	13	1	3	0
AXI,LT	31	25	12	1	3	0
AXI,PT	82	26	12	1	3	0
AXI,LQ	84	27	12	1	3	0
AXI,PQ	85	28	12	1	3	0
TN,LT	91	2	13	1	3	4
TN,PT	92	3	13	1	3	4
TN,CT	93	4	13	1	3	4
TN,LQ	94	5	13	1	3	4
TN,PQ	95	6	13	1	3	4
TN,CQ	96	7	13	1	3	0
TK,LW	101	8	16	1	3	0
TK,PW	102	9	16	1	3	0
TK,CW	103	10	16	1	3	0
TK,LB	104	11	16	1	3	0
TK,PB	105	12	16	1	3	0
TK,CB	106	13	16	1	3	0
SOL,LT	111	14	4	1	3	4
SOL,LW	112	16	4	1	3	4
SOL,PW	113	17	4	1	3	4
SOL,CW	114	18	4	1	3	0

SOL,LB	115	19	4	1	3	4
SOL,PB	116	20	4	1	3	4
SOL,CB	117	21	4	1	3	4
RB	121	34	1	1	0	0
SP,NNS	131	29	3	2	5	6
SP,NGS	132	30	17	2	5	6
SP,NND	133	31	18	2	5	6
SP,NGD	134	32	19	2	5	6
SP,LM	135	33	15	2	5	6
PI,SI	31	1	8	1	0	0
PI,EL	32	1	8	1	0	0
BM,LB	21	1	2	1	0	0
BM,TB	22	1	6	1	0	0
BM,CB	23	1	7	1	0	0
RD	11	1	5	1	0	0

APPENDIX B

Source Code for GEOMETRY Module

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                      C
C        GEOMETRY VER 2.51   06-30-1986    C
C                                      C
C            BOEING AEROSPACE CO.          C
C                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Geometry is a code for producing the geometry tables used by the
C  BUMPER code for evaluating space structures impacted by man made
C  orbital debris or meteoroids. The geometry of the structure is
C  defined by a Supertab universal file which the code reads in. The
C  coordinates of the grid points in the universal file must be in
C  the basic coordinate sytem. The code outputs the threat information,
C  the element id and pid, and each element's surface area. In
C  addition for each threat considered the code outputs a list of the
C  exposed elements and the cosine of the impact angle on each exposed
C  element.
C
C  The code was developed under the NASA contract 'Space Station
C  Integrated Wall Design Guide and Penetration Damage Control'
C  by M.A. Wright and A.R.Coronado.
C
C
C
C  Common Block
C
C  Variable List
C
C    it = current relative threat angle case
C    itype = analysis type , 1-man-made debris, 2-meteoroids
C    nelm = total number of elements
C    npe = number of potentially exposed elements for the current
C          relative threat angle case
C    nt = total number of relative threat angle cases
C
C
C  Array list
C
C    element = global array containing the primary element data as read
C              from Supertab universal file
C
C               1- x coordinate of grid 1
C               2- y     "      "   "  "
C               3- z     "      "   "  "
C               4- x     "      "   "  2
C               5- y     "      "   "  "
C               6- z     "      "   "  "
C               7- x     "      "   "  3
C               8- y     "      "   "  "
C               9- z     "      "   "  "
C              10- a value of the unit normal vector
C              11- b  "    "  "   "    "     "
C              12- c  "    "  "   "    "     "
C              13- x coordinate of the centroid
C              14- y  "         "   "   "
C              15- z  "         "   "   "
C              16- square of the radius enclosing element,origin at cent.
C              17- surface area
C
C   id = global array containing the element id, property id , and grid
C        point id's
```

```
C
C               1- eid
C               2- pid
C               3- idg1
C               4- idg2
C               5- idg3
C
C    iel = global list of the elements containing exposure logical values
C          for the current threat angle
C
C               1- frontside element
C               2- exposed element
C
C    iexp = global list of the potentially exposed element pointers for
C           the current threat angle sorted on the x and y transformed
C           axis
C
C               1- sorted on x tranformed
C               2-   "     " y     "
C
C    point = global array of pointers , indicates where in IEXP an element
C            is located
C
C               1- location in IEXP(1,I)
C               2- location in IEXP(2,I)
C
C    transform = global array containing the element data in the coordinate
C                system parallel to the current threat angle
C
C               1- cosine of the impact angle, measured from the normal
C               2- x transformed coordinate for the centroid
C               3- y    "           "       "   "     "
C               4- z    "           "       "   "     "
C               5- x    "           "       "  grid 1
C               6- y    "           "       "   "  "
C               7- z    "           "       "   "  "
C               8- x    "           "       "   "  2
C               9- y    "           "       "   "  "
C              10- z    "           "       "   "  "
C              11- x    "           "       "   "  3
C              12- y    "           "       "   "  "
C              13- z    "           "       "   "  "
C
C    threat = global array containing the threat data
C
C               1- theta angle, measured from space station velocity
C                  vector in the horizontal plane, radians
C               2- phi angle, measured from the Earth normal, radians
C               3- relative velocity of the threat particle with respect
C                  to the Space Station, km/sec
C               4- probablity of the relative threat occuring
C
C
C  Main Program
C
C  Variable List
C
C    ifile = filename of the Supertab universal file
C    ivar = variable in transform to sort on
C    ofile = output filename
C
C
C
```

```
C
      INCLUDE 'COMMON1.BLK'
C
      CHARACTER*80 IFILE,OFILE
C
C
C  Write header, obtain analysis type and input and output filenames
C
      CALL HEADER ( IFILE,OFILE )
C
C  Open files
C
   10 OPEN (UNIT=2,FILE=IFILE,STATUS='OLD',ERR=20)
C
      GO TO 50
C
C  Error control on open
C
   20 WRITE ( 6,30 )IFILE
   30 FORMAT ( /1X,'UNABLE TO OPEN SUPERTAB UNIVERSAL FILE ',A,
     1         /1X,'SUPERTAB UNIVERSAL FILENAME >' )
      READ ( 5,40 ) IFILE
   40 FORMAT (A)
      GO TO 10
C
   50 OPEN (UNIT=4,FILE=OFILE,STATUS='UNKNOWN',FORM='UNFORMATTED',
     1      ERR=60)
C
      REWIND 4
C
      GO TO 75
C
C  Error control on open
C
   60 WRITE ( 6,70 )OFILE
   70 FORMAT ( /1X,'UNABLE TO OPEN OUTPUT FILE ',A,
     1          1X,'OUTPUT FILENAME >')
      READ ( 5,40 ) OFILE
      GO TO 50
C
C  Create threat array
C
   75 IF ( ITYPE.EQ.1 ) THEN
         CALL DTHREAT
      ELSE
         CALL MTHREAT
      END IF
C
C  Read in Supertab data base
C
      CALL DATA
C
C  Calculate unit vector normals for all elements
C
      CALL NORMAL
C
C  Calculate centroids for all elements
C
      CALL CENTROID
C
C  Calculate the area of each element in global coordinates
C
      CALL AREA
```

```
C
C  Calculate maximum radius enclosing each element
C
      CALL RADIUS
C
C  This section determines hidden/exposed surfaces (elements)
C
C  Initialize NPE to 0
C
      NPE=0
C
      DO 100 I=1,NT
C
          IT=I
C
C  Eliminate all elements whose normal vector points away from the
C  threat  ( backsidc elements )
C
          CALL BACKSIDE
C
C  Transform remaining elements to a coordinate system parallel to the
C  threat
C
          CALL TRANS
C
C  Sort the potentially exposed elements on their transformed x distance
C  from the threat in descending order
C
          IVAR=2
          CALL QSORT ( IVAR )
C
C  Sort the potentially exposed elements on their transformed y distance
C  from the threat in descending order
C
          IVAR=3
          CALL QSORT ( IVAR )
C
C  Generate the element pointers for the sorted lists
C
          DO 80 J=1,NPE
              POINT(1,(IEXP(1,J)))=J
              POINT(2,(IEXP(2,J)))=J
   80     CONTINUE
C
C  Eliminate the elements that are in the shadow of other elements
C
          CALL SHADOW
C
C  Write data to the output file
C
          CALL OUTPUT
C
C  Next threat direction
C
C  Write completed number to the screen
C
      WRITE ( 6,90 ) IT
   90 FORMAT(1X,'THREAT CASE ',I4,' COMPLETED')
C
  100 CONTINUE
C
C  Write out location of output file to the screen
C
```

```
      WRITE (6,110) OFILE
  110 FORMAT (/1X,'OUTPUT LOCATED IN FILE ', A)
C
C  Close files and save
C
      CLOSE ( UNIT = 2 , STATUS = 'KEEP' )
      CLOSE ( UNIT = 4 , STATUS = 'KEEP' )
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE HEADER (IFILE,OFILE)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Header writes the program header to the screen and reads in the
C  supertab universal file name and the output filename. It also
C  determines the analysis type.
C
C
C  Variable list
C
C    answer = character string representing user input
C    ifile = character string for input file name
C    ofile = character string for output file name
C
C
C
      INCLUDE 'COMMON1.BLK'
C
      CHARACTER*80 ANSWER,IFILE,OFILE
C
C  Write header to screen
C
      WRITE ( 6,10 )
10    FORMAT (/1X,'GEOMETRY VER 2.51')
C
C  Determine analysis type, set default to 1 (debris)
C
   15 WRITE ( 6,20 )
   20 FORMAT (/1X,'ANALYSIS TYPE ?',/2X,'1-DEBRIS <CR> ',/2X,
     1         '2-METEOROIDS',/1X,'ANSWER 1 OR 2 >',$)
C
      READ ( 5,30 ) ANSWER
   30 FORMAT (A)
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         ITYPE=1
      ELSE
         READ ( ANSWER(1:80),40 )ITYPE
 40      FORMAT ( BN,I4 )
      END IF
C
C  Check that input was correct
C
      IF ( ITYPE.EQ.1 .OR. ITYPE.EQ.2 ) THEN
         CONTINUE
      ELSE
         WRITE ( 6,50 )
 50      FORMAT ( /1X,'INCORRECT INPUT' )
         GO TO 15
      END IF
C
C  Read Supertab universal filename, or set to default to station.uni
C
      WRITE ( 6,60 )
   60 FORMAT (/1X,'SUPERTAB UNIVERSAL FILENAME (CR=STATION.UNI) >',$)
      READ ( 5,30 )IFILE
      IF ( IFILE(1:1).EQ.' ' )IFILE='STATION.UNI'
C
```

```
C  Read output file name, or set to default to station.gem
C
      WRITE ( 6,70 )
   70 FORMAT (/1X,'OUTPUT FILENAME (CR=STATION.GEM) >',$)
      READ ( 5 , 30 )OFILE
      IF (OFILE(1:1).EQ.' ')OFILE='STATION.GEM'
C
C  Finished
C
      RETURN
C
      END
```

```
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
      SUBROUTINE DTHREAT
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c
c  Dthreat builds the Threat array for man-made orbital debris
c  analysis. The code models cnly relative threats between -90 and 90
c  degrees in the horizontal plane measured from the space station
c  velocity vector. Orbital mechanics show that only these relative
c  threats can occur.
c
c  The code calculates the velocity of the debris particle relative to
c  the space station. The probability of the relative threat occuring
c  is then derived from the NASA JSC-20001 velocity distribution. The
c  orbital altitude is assumed to be 500 km.
c
c  The code requires that the relative velocity distribution be defined
c  in the user file DEB.VEL . The file is a list of third order curves
c  defining the various ranges of the distribution. There can be up
c  to 20 ranges. The data is read in free field format. The ranges must
c  be in ascending order. The record format is as follows :
c
c         record 1-20 - nr,rmin,rmax,c1,c2,c3,c4
c
c         where
c
c          c1-c4 = third order curve constants
c                  p(vr)= c1 + c2*vr + c3*vr**2 + c4*vr**3
c          nr = range number
c          rmax = max relative velocity in range nr
c          rmin = min relative velocity in range nr
c
c
c
c  note: for variables in the common block see the main listing
c
c  Variable List
c
c    answer = character string representing user input
c    cp1 = cumulative relative threat angle probability at angle T1
c    cp2 =    "       "        "      "       "          "    "  T2
c    c1,c2,c3,c4 = third order curve fit constants for the relative
c                  velocity distribution at range nr
c    ic = counter
c    ip1 = pointer for angle T1 in the DATA array
c    ip2 =   "     "   "  T2  "  "   "    "
c    ip12 = pointer for the next angle after ip1
c    ip22 =   "     "   "   "    "     "   ip2
c    ncal = number of calculations for the DATA arrays
c    nr = current range number for the relative velocity distribution
c    pra = probability densisty of the current relative threat
c    prv =   "          "        "   "     "       "     velocity
c    prob = probability of the current threat (+/- tinc/2) occuring
c    phi = relative threat angle measured from the Earth normal,radians
c    rat1,rat2 = intermediate linear interpolation variables
c    sum = sum of the current list program is evaluating
c    tc = theta increment used in the DATA array calculations, radians
c    thc = current theta , used in DATA array calculations, radians
c    theta = relative threat angle measured from the space station
c            velocity vector in the horizontal plane, radians
```

```
C    tinc = theta increment in the THREAT array, radians
C    t1 = theta plus tinc/2 ,radians
C    t2 =   "   minus tinc/2, radians
C    vmax = maximum relative velocity in VDIST, km/sec
C    vmin = minimum   "         "        "    "    "
C    vr = relative velocity of the debris particle with respect to the
C         space station, km/sec
C    vs = space station orbital velocity, km/sec
C    v11,v12,v21,v22 = intermediate linear interpolation variables
C    xrmax = maximum relative velocity in range nr, km/sec
C    xrmin = minimum   "          "       "   "   "   "
C
C
C  Array List
C
C    data = working list of the relative threat angle probability
C           densisty and the cumulative relative angle probability.
C           Used to calculate the Threat array.
C    threat = global array containg the threat information, theta,phi,
C            vr,prob
C    vdist = array containg third order curve constants describing the
C            the NASA JSC 20001 relative velocity distribution
C
C
C
C
      INCLUDE 'COMMON1.BLK'
C
      DIMENSION VDIST(6,20),DATA(2,2000)
C
      CHARACTER*90 ANSWER
C
      PARAMETER (PI=3.1415926536)
C
C  Set the space station velocity
C
      VS=7.50D0
C
C  Set Phi equal to a constant ( PI/2 )
C
      PHI=PI/2.0
C
C  Set up the working array variables
C
      NCAL=1000
      TC=PI/2.0/NCAL
C
C  Read in the number of uniform threats, set default to 45
C
   40 WRITE ( 6,50 )
   50 FORMAT ( /1X,'NUMBER OF UNIFORM DEBRIS THREATS  (CR=45) >',$)
      READ ( 5,60 ) ANSWER
   60 FORMAT (A)
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         NT=45
      ELSE
         READ ( ANSWER(1:90),70 ) NT
   70    FORMAT ( BN,I6 )
      END IF
C
C  Check that the number of threats is less than 200
C
```

```
      IF ( NT.GT.200 ) THEN
         WRITE ( 6,80 ) NT
   80    FORMAT ( /1X,'NUMBER OF THREATS (',I4,') IS OUTSIDE OF RANGE')
         GO TO 40
      END IF
C
C  Calculate the Theta increment in the threat array
C
      TINC=PI/NT
C
C  Open the DEB.VEL file and read in the relative velocity distribution
C  data
C
      OPEN ( UNIT=7,FILE='DEB.VEL',STATUS='OLD',ERR=100 )
C
      GO TO 200
C
C  Error control for open
C
  100 WRITE ( 6,110 )
  110 FORMAT ( /1X,'DEBRIS VELOCITY DISTRIBUTION  FILE DEB.VEL WAS',
     1              ' NOT FOUND'/' FILENAME ? >')
      READ ( 5,60 ) ANSWER
C
      IF ( ANSWER(1:2).EQ.'  ' ) GO TO 100
C
      OPEN ( UNIT=7,FILE=ANSWER,STATUS='OLD',ERR=100 )
C
C  Read the data, counting the number of ranges read
C
  200 IC=0
      DO 225 I=1,20
         READ ( 7,*,END=250 )NR,(VDIST(J,I),J=1,6)
         IC=IC+1
  225 CONTINUE
C
  250 CLOSE ( UNIT=7,STATUS='KEEP' )
C
C  Determine the minimum and maximum allowable relative velocities
C
      VMIN=VDIST(1,1)
      VMAX=VDIST(2,IC)
C
C  Intialize the range pointer to 1
C
      NR=1
C
C  Calculate the relative threat angle probability density for the
C  working array
C
      DO 400 I=1,NCAL+1
C
C  Set the Theta angle and calculate the relative velocity. From
C  orbital mechanics it is known that the velocity of the station is
C  approximately equal to the velocity of the debris particle. Vr is
C  related to vs through vector addition.
C
         THC=(I-1)*TC
         VR=2.0*VS*ABS(COS(THC))
C
C  Check that vr is in the range of the relative velocity distribution
C  data
C
```

```
    IF ( VR.LT.VMIN .OR. VR.GT.VMAX ) THEN
      WRITE ( 6,275 )VR
 275  FORMAT ( /1X,'IMPACT VELOCITY OUTSIDE OF ALL RANGES ',/
  1             1X,'VR=',E12.5)
      STOP
    END IF
C
C Get the current relative velocity range
C
 300    XRMIN=VDIST(1,NR)
    XRMAX=VDIST(2,NR)
C
C Check that this is the correct range, if not increment accordingly
C being sure to not step outside of VDIST array
C
      IF(VR.LT.XRMIN)THEN
      NR=NR-1
C
      IF ( NR.LT.1 ) THEN
       WRITE ( 6,350 )
 350   FORMAT ( /1X,'CANNOT FIND CORRECT VELOCITY RANGE ' )
       STOP
      END IF
C
      GO TO 300
    END IF
C
    IF(VR.GT.XRMAX)THEN
      NR=NR+1
C
      IF ( NR.GT.IC ) THEN
       WRITE ( 6,350 )
       STOP
      END IF
C
      GO TO 300
    END IF
C
C Get curve fit constants
C
    C1=VDIST(3,NR)
    C2=VDIST(4,NR)
    C3=VDIST(5,NR)
    C4=VDIST(6,NR)
C
C Calculate the relative velocity probability density
C
    PRV=C1+C2*VR+C3*VR**2.0+C4*VR**3.0
C
C Calculate the associated relative threat angle probability density,
C it is the relative velocity probability density multiplied by the
C absolute value of the derivative of the equation relating the two.
C
    PRA=PRV*2.0*VS*ABS(SIN(THC))
C
C Store in the DATA array
C
    DATA(1,I)=PRA
C
C Next threat
C
 400 CONTINUE
C
```

```
C  Use a trapezoid approximation to determine the area under the
C  probability density curve. The running sum of this value is the
C  cumulative probability. Store the cumulative probability in the
C  DATA array.
C
      DATA(2,1)=0.0D0
C
      SUM=0.0D0
      DO 500 I=1,NCAL
         PROB=0.50*TC*(DATA(1,I)+DATA(1,(I+1)))
         SUM=SUM+PROB
         DATA(2,(I+1))=SUM
  500 CONTINUE
C
C  Use the DATA array to determine the THREAT array
C
      DO 700 I=1,NT
C
C  Set Theta and calculate the relative velocity
C
         THETA=(TINC/2.0+(I-1)*TINC)-PI/2.0
         VR=2.0*VS*ABS(COS(THETA))
C
C  Determine the location of the nearest angle in the data array
C  to the theta that is still less than theta
C
         IP=ABS(THETA/TC)+1
C
C  Set the threat angle increment , T1 to T2
C
         T1=THETA-TINC/2.0
         T2=THETA+TINC/2.0
C
C  Determine the location of nearest angle to T1 & T2 in the Data
C  array that is still less than T1 & T2
C
         IP1=ABS(T1/TC)+1
         IP2=ABS(T2/TC)+1
C
C  Determine the location of the next largest angle in the Data array
C  for T1 and T2. Checking that it is not outside the array.
C
         IP12=IP1+1
         IF ( IP12.GT.NCAL+1 ) IP12=NCAL+1
         IP22=IP2+1
         IF ( IP22.GT.NCAL+1 ) IP22=NCAL+1
C
C  Estimate the probability of the threat occuring by using the cumulative
C  probability values in the Data array and linear interpolation.
C
C  Get the cumulative probability data from the Data array
C
         V11=DATA(2,IP1)
         V12=DATA(2,IP12)
         V21=DATA(2,IP2)
         V22=DATA(2,IP22)
C
C  Perform the interpolation
C
         RAT1=ABS(T1/TC-INT(T1/TC))
         RAT2=ABS(T2/TC-INT(T2/TC))
C
         CP1=RAT1*(V12-V11)+V11
         CP2=RAT2*(V22-V21)+V21
C
C  For angle increments  spanning the 0.0 angle add the cumlative values.
C  Else subract the values.
C
         IF ( IP.LE.1 ) THEN
            PROB=ABS(CP1+CP2)
         ELSE
            PROB=ABS(CP2-CP1)
         END IF
C
C  Store the data in the Threat array
C
         THREAT(1,I)=THETA
         THREAT(2,I)=PHI
         THREAT(3,I)=VR
         THREAT(4,I)=PROB
C
C  Next threat
C
  700 CONTINUE
C
C  Normalize the probabilty data in the threat array to 1.0
C
      SUM=0.0D0
      DO 800 I=1,NT
         SUM=SUM+THREAT(4,I)
  800 CONTINUE
C
      DO 950 I=1,NT
         THREAT(4,I)=THREAT(4,I)/SUM
  850 CONTINUE
C
C  Finished
C
      RETURN
C
      END
```

```
c
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
      SUBROUTINE MTHREAT
c
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c
c  Mthreat builds the Threat array for meteoroid analysis. The code
c  models the relative threat as a hemisphere broken into approximately
c  equal area square elements ( similar to an Igloo ). A given relative
c  threat is determined by calculating the Theta and Phi angles mesured
c  from the origin to the given element. The probability of this
c  relative threat occuring is equal to the ratio of the element's
c  surface area to the total surface area of the hemisphere.
c
c  The probability is further modified to account for the orbital motion
c  of the space station. The velocity of the meteoroid relative to the
c  space station is calculated as well as the flux focusing factor. Ted
c  Hansen developed the flux focusing factor.
c
c  As a first order approximation the meteoroids are assumed to all have
c  a absolute velocity equal to the average meteoroid velocity. This
c  ignores the actual velocity distribution but appears to give reasonable
c  results while greatly simplifying the analysis.
c
c  All the calculations are done in double precision but stored as single
c  precision in the Threat array.
c
c  Note:  for variables included in the common block see main listing
c
c  Variable List
c
c    alpha = angle between the space station velocity vector and the
c            relative threat vector, radians
c    alt = altitude of the space station less 100km atmosphere, km
c    answer = character string representing user input to a given question
c    ar = ratio of a given element's surface area to the hemisphere surface
c         area
c    chi = pi - gamma, radians
c    del = the delta phi angle, equal for each ring in the hemisphere,
c          radians
c    dphi = the angle measured from the z axis to the relative threat vector
c           double precision, radians
c    dpi = double precision pi
c    dtheta = the angle measured from the space station velocity vector
c             to the relative threat vector, double precision , radians
c    gamma = half angle of the shaded cone , radians
c    h = vertical height of a given ring
c    l1 = intermediate variable, radians
c    l2 =     "           "          "
c    ic = counter
c    prob = probability of a given relative threat occuring
c    re = earth's radius (including 100 km  atmosphere), km
c    sf = portion of space station exposed to meteoroids
c    sum = running sum of the probabilities
c    tinc = theta increment in a given ring, radians
c    tmf = flux focusing factor
c    vm = velocity of the meteoroid, km/sec
c    vr = velocity of the meteoroid relative to the space station , km/sec
c    vs = orbital velocity of the space station, km/sec
c    vstar = intermediate variable, km/sec
```

```
C
C
C Array list
C
C  area =  area of the elements in a ring
C  dthreat = threat array in double precision
C  nel = number of elements in a ring
C  threat = array containing the relative theta & phi angles,
C          the realtive velocity and the relative threat probability
C
C
      INCLUDE 'COMMON1.BLK'
C
      DIMENSION NEL(50)
C
      REAL*8 DPI
C
      PARAMETER ( DPI=3.141592653589793238D0 )
C
      CHARACTER*90 ANSWER
C
      REAL*8 ALPHA,ALT,AR,CHI,DEL,DPHI,DTHETA,GAMMA,H,L1,L2,PROB,RE
     1       SF,SUM,TMF,VM,VR,VS,VSTAR,AREA(50),DTHREAT(4,1000)
C
C Set the station and meteoroid velocity
C
      VS=7.50D0
      VM=20.0D0
C
C Set the Earth radius and Space Station altitude
C
      RE=6478.0D0
      ALT=500.0D0-100.0D0
C
C Read in the threat case to be run
C
   40 WRITE ( 6,50 )
   50 FORMAT ( 1X,'NUMBER OF UNIFORM METEOROID THREATS ?',
     1              /5X,'1 - 84',/5X,'2 - 146<CR>',/5X,'3 - 232',
     2              /5X,'4 - 329',/1X,'ANSWER (1-4) >',$)
C
      READ ( 5,60 ) ANSWER
   60 FORMAT (A)
C
      IF ( ANSWER(1:2).EQ.'  ' ) THEN
         IANS=2
      ELSE
         READ ( ANSWER(1:80),70 ) IANS
   70    FORMAT ( BN,I6 )
      END IF
C
C Check if the threat case is within the given range
C
      IF ( IANS.GT.4 .OR. IANS.LT.1 ) THEN
         WRITE ( 6,80 )
   80    FORMAT ( '0','ANSWER OUTSIDE OF RANGE ' )
         GO TO 40
      END IF
C
C Determine the half angle of the shaded cone,
C equation is from JSC-30000
C
      GAMMA=DASIN(RE/(RE+ALT))
      CHI=DPI-GAMMA
C
C  Calculate the number of uniform rings in the hemisphere, and the
C  delta phi angle for each ring
C
      NR=6+(IANS-1)*2
      DEL=CHI/NR
C
C  Determine the number elements and their area in the first ring
C
      NEL(1)=IDNINT(2.0D0*DPI/DEL)
      H=DSIN(DEL)
      AREA(1)=2.0D0*DPI*H/NEL(1)
C
C  Do the same for all the other rings holding the area approximately
C  equal.
C
      DO 100 I=2,NR
C
         H=ABS(DCOS(CHI-I*DEL)-DCOS(CHI-(I-1)*DEL))
         NEL(I)=IDNINT(2.0D0*DPI*H/AREA(1))
         AREA(I)=2.0D0*DPI*H/NEL(I)
C
  100 CONTINUE
C
C  Determine the number of elements ( equal to the number of  threats )
C
      NT=0
      DO 200 I=1,NR
         NT=NT+NEL(I)
  200 CONTINUE
C
C  Each element represents a given relative threat direction. For each
C  element determine the location of it's C.G. . Also calculate the
C  relative meteoroid velocity for this threat and the flux focusing
C  factor. Store the results in the Threat array.
C
      IC=0
C
      DO 400 I=1,NR
C
C  Calculate the theta increment in the ring
C
         TINC=2.0D0*DPI/NEL(I)
C
C  Calculate phi for the C.G., constant for each element in the ring
C
         L1=DPI/2.0+(I-1)*DEL-CHI
         L2=L1+DEL
         DPHI=DACOS(0.50D0*(DSIN(L1)+DSIN(L2)))
C
C  Calculate the probability of the threat occuring, constant for each
C  element in the ring
C
         AR=AREA(I)/4.0D0/DPI
C
C  Evaluate each element
C
         DO 300 J=1,NEL(I)
C
            IC=IC+1
C
C  Determine the theta of the C.G.
```

```
C
           DTHETA=TINC/2.0D0+(J-1)*TINC
C
C  Calculate the relative meteoroid velocity and the flux focusing factor
C
           ALPHA=DACOS(DSIN(DPHI)*DCOS(DTHETA))
           VSTAR=DSQRT(VM**2-VS**2*(DSIN(ALPHA)**2))
           VR=VSTAR+VS*DCOS(ALPHA)
C
           TMF=VR**3/VSTAR/VM**2
C
C  Apply the focusing factor to the probability
C
           PROB=AR*TMF
C
C  Store the values in the Threat array
C
           DTHREAT(1,IC)=DTHETA
           DTHREAT(2,IC)=DPHI
           DTHREAT(3,IC)=VR
           DTHREAT(4,IC)=PROB
C
  300   CONTINUE
C
  400 CONTINUE
C
C  Store Dthreat array in Threat
C
      DO 600 I=1,NT
         THREAT(1,I)=DTHREAT(1,I)
         THREAT(2,I)=DTHREAT(2,I)
         THREAT(3,I)=DTHREAT(3,I)
         THREAT(4,I)=DTHREAT(4,I)
  600 CONTINUE
C
C  Finished
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE DATA
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Data reads in the supertab universal file and places the
C  element and nodal data in the global element array.
C
C
C  note: for variables in common block see main listing
C
C  variable list
C
C    dline = character string representing all the data contained
C            on one line of the supertab universal file
C    eid = element identification
C    ig = grid id
C    igk = id of grid point in location k
C    ival = value contained in the first 6 spaces of dline
C    igl = id for element grid point 1
C    ig2 =  "   "     "     "     "   2
C    ig3 =  "   "     "     "     "   3
C    k = grid location pointer
C    kh = largest value of k
C    kl = smallest value of k
C    kp = previous value of k
C    ngrids = number of grids
C    pid = element property id
C    x = global x position , Meters
C    y =   "    y   "      ,   "
C    z =   "    z   "      ,   "
C
C   array list
C
C    grid = working array containing grid point locations in global
C           coordinate system
C    idg = working array containing grid point id's
C
C
C
      CHARACTER*90 DLINE
C
      INCLUDE 'COMMON1.BLK'
C
      DIMENSION GRID(3,ISIZE*3)
C
      INTEGER*4 IG1,IG2,IG3,IG,IGK,IVAL,IDG(ISIZE),K,KH,KL,EID,PID
C
C  Intialize counters
C
      NELM = 0
      NGRIDS = 0
C
C  Read line of data from Supertab file , if end of file is reached
C  continue with processing
C
   10 READ ( 2,20,END=100 )DLINE
   20 FORMAT ( A )
C
C  Read first 6 characters of dline as an integer , if error then read
```

```
C  next line
C
      READ (DLINE(1:6),30,ERR=10)IVAL
   30 FORMAT(BN,I6)
C
C  Check if ival = -1 , which indicates the start of a dataset, if not
C  read another line until -1 is found
C
      IF ( IVAL.NE.-1 ) GO TO 10
C
C  -1 found , read next line of data, then read first six characters
C  if error on read, then read until end of dataset is found
C
      READ ( 2,20 ) DLINE
      READ ( DLINE(1:6),30,ERR=90 ) IVAL
C
C  First 6 characters are the dataset id, check if this is the nodal data
C
      IF ( IVAL.EQ.15 ) THEN
C
C  This is the nodal data , read data and look for the -1
C  that indicates the end the dataset has been reached
C
   40     READ ( 2,20 ) DLINE
          READ ( DLINE(1:6),30 ) IVAL
C
C  Check if bottom of dataset , if so start looking for next dataset
C
          IF ( IVAL.EQ.-1 ) GO TO 10
C
C  If not  read grid id and location
C
          READ ( DLINE(1:90),50 )IG,X,Y,Z
   50     FORMAT ( BN,I10,30X,3E13.5 )
C
C  Increment grid counter
C
          NGRIDS=NGRIDS+1
C
C  Place data in appropriate array
C
          IDG(NGRIDS)=IG
          GRID(1,NGRIDS)=X
          GRID(2,NGRIDS)=Y
          GRID(3,NGRIDS)=Z
C
C  Read next data line
C
          GO TO 40
C
      ELSE
C
C  Is this the element dataset ?
C
          IF ( IVAL.EQ.71 ) THEN
C
C  It is the element, read next line of data, coonstantly checking for
C  end of dataset
C
   60         READ ( 2,20 ) DLINE
              READ ( DLINE(1:6),30 )IVAL
              IF ( IVAL.EQ.-1 ) GO TO 10
C
```

```
C  Read eid and pid
C
               READ ( DLINE(1:90),70 )EID,PID
   70          FORMAT ( BN,I10,20X,I10 )
C
C  Read next line, and read ig1,ig2,ig3 off it
C
               READ ( 2,20 ) DLINE
               READ ( DLINE(1:90),80 )IG1,IG2,IG3
   80          FORMAT( BN,3I10 )
C
C  Increment element counter
C
               NELM=NELM+1
C
C  Place data in appropriate array
C
               ID(1,NELM)=EID
               ID(2,NELM)=PID
               ID(3,NELM)=IG1
               ID(4,NELM)=IG2
               ID(5,NELM)=IG3
C
C  Read in next data line
C
               GO TO 60
C
            ELSE
C
C  It isn't the  nodal or element dataset , read lines unitl bottom of
C  dataset is found, or end of file is found , if error on read read next
C  line
C
   90          READ ( 2,20 )DLINE
               READ ( DLINE(1:6),30,ERR=90,END=100 )IVAL
               IF ( IVAL.NE.-1 ) GO TO 90
               GO TO 10
C
            END IF
C
         END IF
C
C  Check if element data was read in, if not write error message and stop
C
  100    IF ( NELM.EQ.0 ) THEN
            WRITE ( 6,110 )
  110       FORMAT ( /1X,'NO ELEMENT DATA WAS READ IN ' )
            STOP
         END IF
C
C  Check if nodal data was read in , if not write error message and stop
C
         IF ( NGRIDS.EQ.0 ) THEN
            WRITE ( 6,120 )
  120       FORMAT ( /1X,'NO NODAL DATA WAS READ IN ' )
            STOP
         END IF
C
C  Place information in global element array
C
C  Loop thru all the elements
C
         DO 400 I=1,NELM
```

```
C
C  Loop thru 3 grid points
C
            DO 300 J=1,3
C
C  Initialize grid point  location pointers
C
                KH=NGRIDS+1
                KL=1
                KP=0
C
C  Get grid id
C
                IG=ID((3+(J-1)),I)
C
C  Make guess of grid location using binary search technique
C
  210           K=(KL+KH)/2
C
C  Check if same location is being guessed again, if it is write
C  error message and stop
C
                IF ( KP.EQ.K ) THEN
                   WRITE ( 6,220 ) IG
  220              FORMAT(/1X,'DATA NOT FOUND FOR NODE ',I5 )
                   STOP
                END IF
C
C  Get grid id for location k
C
                IGK=IDG(K)
C
C  If this is the correct location put data in array element
C
                IF ( IGK.EQ.IG ) THEN
C
C  Loop thru 3 coordinates
C
                     DO 250 L=1,3
                         ELEMENT((L+(J-1)*3),I)=GRID(L,K)
  250                CONTINUE
C
C  If not , reset location limits , guess again
C
                ELSE
C
C  Guess too low in list , reset lower bound
C
                   IF ( IGK.LT.IG ) KL=K
C
C  Guess too high in list , reset upper bound
C
                   IF ( IGK.GT.IG ) KH=K
C
C  Reset holder
C
                   KP=K
C
C  Guess again
C
                   GO TO 210
C
                END IF
```

```
C
C  Next  grid point
C
   300    CONTINUE
C
C  Next element
C
   400 CONTINUE
C
C
C  finished , return
C
C
       RETURN
C
       END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE NORMAL
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Normal calculates the unit normal vector for each element and stores
C  it in the element array.
C
C  variable list
C
C     an =  i component of the normal vector
C     a12 = "    "        "  "   12       "
C     a13 = "    "        "  "   13       "
C     bn =  j component of the normal vector
C     b12 = "    "        "  "   12       "
C     b13 = "    "        "  "   13       "
C     cn =  k component of the normal vector
C     c12 = "    "        "  "   12       "
C     c13 = "    "        "  "   13       "
C     r = length of the normal vector
C
C
C
      INCLUDE 'COMMON1.BLK'
C
C  Calculate unit normal vector for each element
C
      DO 100 I=1,NELM
C
C  Calculate vector from grid point 1 to grid point 2
C
         A12=ELEMENT(4,I)-ELEMENT(1,I)
         B12=ELEMENT(5,I)-ELEMENT(2,I)
         C12=ELEMENT(6,I)-ELEMENT(3,I)
C
C  Calculate vector from grid point 1 to grid point 3
C
         A13=ELEMENT(7,I)-ELEMENT(1,I)
         B13=ELEMENT(8,I)-ELEMENT(2,I)
         C13=ELEMENT(9,I)-ELEMENT(3,I)
C
C  Calculate the normal vector, it is equal to the cross product of vetor
C  12 and vector 13
C
         AN=(B12*C13)-(C12*B13)
         BN=(C12*A13)-(A12*C13)
         CN=(A12*B13)-(B12*A13)
C
C  Calculate the length of the normal vector
C
         R=SQRT(AN**2+BN**2+CN**2)
C
C  The unit normal vector is equal to the normal vector divided by it's
C  length, place data in element array
C
         ELEMENT(10,I)=AN/R
         ELEMENT(11,I)=BN/R
         ELEMENT(12,I)=CN/R
C
C  Next element
```

```
C
  100 CONTINUE
C
C  Return
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCVCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE CENTROID
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  Centroid determines the location of the centroid for each element,
C  and stores it in the element array.
C
C  Variable list
C
C   xc = x location of the centroid
C   yc = y location of the centroid
C   zc = z location of the centroid
C
C  array list
C
C   x = array containing x values of the grid points
C   y =  "        "      y   "    "   "    "    "
C   z =  "        "      z   "    "   "    "    "
C
C
      DIMENSION X(3),Y(3),Z(3)
C
      INCLUDE 'COMMON1.BLK'
C
C  Determine c.g. for each element
C
      DO 100 I=1,NELM
C
C  Get grid point data from element array
C
         DO 50 J=1,3
C
            X(J)=ELEMENT((1+(J-1)*3),I)
            Y(J)=ELEMENT((2+(J-1)*3),I)
            Z(J)=ELEMENT((3+(J-1)*3),I)
C
C  Next grid point
C
   50    CONTINUE
C
C  C.G. location is average of 3 grid point locations
C
         XC=(X(1)+X(2)+X(3))/3.0
         YC=(Y(1)+Y(2)+Y(3))/3.0
         ZC=(Z(1)+Z(2)+Z(3))/3.0
C
C  Put data in element array
C
         ELEMENT(13,I)=XC
         ELEMENT(14,I)=YC
         ELEMENT(15,I)=ZC
C
C  Next element
C
  100 CONTINUE
C
C  Return
C
      RETURN
C
```

```
0082            END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE AREA
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  Area calculates the surface area of each element.
C
C
C  Variable list
C
C   atri = surface area of the element , sq-meters
C   d1 = intermediate variable
C   d2 =      "         "
C   d3 =      "         "
C
C  Array list
C
C   x = working array containing the grid point x coordinates
C   y =   "       "       "       "    "    "   y    "
C   z =   "       "       "       "    "    "   z    "
C
C
      INCLUDE 'COMMON1.BLK'
C
      DIMENSION X(3),Y(3),Z(3)
C
C  For each element calculate the surface area.
C
      DO 100 I=1,NELM
C
C  Get the location of the 3 grid points in global coordinates.
C
         DO 50 J=1,3
C
            X(J)=ELEMENT((1+(J-1)*3),I)
            Y(J)=ELEMENT((2+(J-1)*3),I)
            Z(J)=ELEMENT((3+(J-1)*3),I)
C
   50    CONTINUE
C
C  Calculate the surface area using the equation from the CRC Math Handbook
C
         D1=Y(1)*(Z(2)-Z(3))-Z(1)*(Y(2)-Y(3))+(Y(2)*Z(3)-Z(2)*Y(3))
C
         D2=Z(1)*(X(2)-X(3))-X(1)*(Z(2)-Z(3))+(Z(2)*X(3)-X(2)*Z(3))
C
         D3=X(1)*(Y(2)-Y(3))-Y(1)*(X(2)-X(3))+(X(2)*Y(3)-Y(2)*X(3))
C
         ATRI = 0.50 * SQRT( D1**2 + D2**2 + D3**2)
C
C  Save the area in the element array
C
         ELEMENT(17,I)=ATRI
C
C  Next element
C
  100 CONTINUE
C
C
      RETURN
```

```
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE RADIUS
C
CCCCC6CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Radius calculates the maximum radius that will contain the three grid
C  points of each triangular element, with the center located at the
C  centroid.
C
C  Variable list
C
C    a(k) = component 1 of vector from c.g. to grid k of element i
C    b(k) = component 2 of vector from c.g. to grid k of element i
C    c(k) = component 3 of vector from c.g. to grid k of element i
C    i = counter from 1 thru number of elements (nelm)
C    j = counter from 1 thru 3 nodes per element
C    k = counter from 1 thru 3 components per vector
C    l = counter from 1 thru 3 vectors per element
C    r(i) = length of vectors from c.g. to grid
C    rmax = maximum radius (squared) that contains all three nodes
C    x(j) = x coordinate of Ith element grid j
C    y(j) = y coordinate of Ith element grid j
C    z(j) = z coordinate of Ith element grid j
C    xc =  x coordinate of Ith element c.g.
C    yc =  y coordinate of Ith element c.g.
C    zc =  z coordinate of Ith element c.g.
C
C
      INCLUDE 'COMMON1.BLK'
C
      DIMENSION A(3),B(3),C(3),R(3),X(3),Y(3),Z(3)
C
      DO 10 I=1,NELM
C
C  Read coordinates of c.g. of element I
C
         XC=ELEMENT(13,I)
         YC=ELEMENT(14,I)
         ZC=ELEMENT(15,I)
C
            DO 20 J=1,3
C
C  Read coordinates of three grids of element I
C
               X(J)=ELEMENT((1+(J-1)*3),I)
               Y(J)=ELEMENT((2+(J-1)*3),I)
               Z(J)=ELEMENT((3+(J-1)*3),I)
C
   20       CONTINUE
C
            DO 30 K=1,3
C
C  Calculate vector from c.g. to each grid
C
               A(K)=XC-X(K)
               B(K)=YC-Y(K)
               C(K)=ZC-Z(K)
C
   30        CONTINUE
C
```

```
             DO 40 L=1,3
C
C  Calculate length (squared) of each vector from c.g. to each grid
C
                 R(L)=A(L)**2+B(L)**2+C(L)**2
C
   40        CONTINUE
C
C  Determine length of longest vector which will then be radius
C
         RMAX=AMAX1( R(1),R(2),R(3) )
C
         ELEMENT( 16,I )=RMAX
C
   10 CONTINUE
C
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE BACKSIDE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C   Backside eliminates the elements not exposed to the relative threat
C   by virtue of the fact that their unit normal vector does not point
C   at the threat. It also generates the list of potentially exposed
C   elements.
C
C
C  Variable list
C
C    a  = i component of the unit threat vector
C    an = "    "       "  "  "    normal   "
C    b  = j    "       "  "  "    threat   "
C    bn = "    "       "  "  "    normal   "
C    c  = k    "       "  "  "    threat   "
C    cn = "    "       "  "  "    normal   "
C    dp = dot product of the unit normal vector and the unit threat
C         vector
C    ipx = element location in row 1 of the IEXP array
C    ipy =    "        "     "  "  2 "   "     "  "
C    phi = relative threat angle measured from the Earth normal, radians
C    theta = relative threat angle measured from the station velocity
C            vector in the horizontal plane, radians
C
C
C
      INCLUDE 'COMMON1.BLK'
C
C  For the first angle intialize the IEL, IEXP, & POINT arrays and
C  the NPE counter
C
      IF ( IT.NE.1 ) GO TO 20
C
      DO 10 I=1,NELM
C
         IEL(1,I)=.TRUE.
         IEL(2,I)=.TRUE.
         IEXP(1,I)=0
         IEXP(2,I)=0
         POINT(1,I)=0
         POINT(2,I)=0
C
   10 CONTINUE
C
      NPE=0
C
C  Get the threat angles from the threat array
C
   20 THETA=THREAT(1,IT)
      PHI=THREAT(2,IT)
C
C  Calculate the unit vector that points at the threat direction,
C  ( threat vector )
C
      A=SIN(PHI)*COS(THETA)
      B=SIN(PHI)*SIN(THETA)
      C=COS(PHI)
```

```
C
C   For each element , check if it is a backside element
C
      DO 100 I=1,NELM
C
C   Get the unit normal vector from the element array
C
         AN=ELEMENT(10,I)
         BN=ELEMENT(11,I)
         CN=ELEMENT(12,I)
C
C   Calculate the dot product of the unit normal vector and the unit
C   threat vector
C
         DP=A*AN+B*BN+C*CN
C
C   Save the dot product in the transform array
C
         TRANSFORM(1,I)=DP
C
C   For the first angle case
C
         IF ( IT.EQ.1 ) THEN
C
C   If the dot product is < 0 , then the element is a backside element
C
            IF(DP.LE.0.01)THEN
C
C   It is backside , change IEL to false
C
              IEL(1,I)=.FALSE.
              IEL(2,I)=.FALSE.
C
            ELSE
C
C   It is potentially exposed , increase NPE by 1 and place I in IEXP
C   location NPE
C
              NPE=NPE+1
              IEXP(1,NPE)=I
              IEXP(2,NPE)=I
C
            END IF
C
C   For all other cases , maintain the relative order of the IEXP list
C   by adding exposed elements at the bottom and removing backside elements
C   and shifting the list up . This will decrease the time required for
C   sorting later in the code.
C
         ELSE
C
C   If the dot product is < 0 then the element is a backside element
C
            IF(DP.LE.0.01)THEN
C
C   It is a backside element , check if it was one in the previous case
C
              IF ( IEL(1,I) ) THEN
C
C   It was not a backside element in the previous case, change IEL to
C   false
C
                IEL(1,I)=.FALSE.
```

```
               IEL(2,I)=.FALSE.
C
C  Now, remove element I from the IEXP list , the list is not in order
C
C
C  Get element I location in IEXP from the pointer array
C
               IPX=POINT(1,I)
               IPY=POINT(2,I)
C
C  Remove element I and shift the list
C
C  Shift the x list first, and reset the pointer list
C
               DO 40 J=IPX,NPE
C
                  IEXP(1,J)=IEXP(1,(J+1))
                  POINT(1,IEXP(1,J+1))=J
C
   40          CONTINUE
C
C  Shift the y list, and reset the pointer list
C
               DO 45 J=IPY,NPE
C
                  IEXP(2,J)=IEXP(2,(J+1))
                  POINT(2,IEXP(2,J+1))=J
C
   45          CONTINUE
C
C  Reduce the number of potentially exposed elements by 1
C
               NPE=NPE-1
C
C  Reset pointer for the removed element to 0
C
               POINT(1,I)=0
               POINT(2,I)=0
C
             END IF
C
C  The element was a backside element in the previous case ,
C  no need to do anything
C
           ELSE
C
C  The element is potentially exposed, check if it was exposed in the
C  previous case
C
             IF ( IEL(1,I) ) THEN
C
C  It was exposed in the previous case , make IEL true
C
                IEL(1,I)=.TRUE.
                IEL(2,I)=.TRUE.
C
             ELSE
C
C  It wasn't exposed previously, change IEL to true, increase NPE by
C  1, add element I to IEXP at location NPE, and save location in the
C  pointer array
C
                IEL(1,I)=.TRUE.
```

```
                  IEL(2,I)=.TRUE.
C
                  NPE=NPE+1
C
                  IEXP(1,NPE)=I
                  IEXP(2,NPE)=I
C
                  POINT(1,I)=NPE
                  POINT(2,I)=NPE
C
              END IF
C
            END IF
C
          END IF
C
C  Next element
C
  100 CONTINUE
C
C  Check if number of exposed elements is 0 , if so write message to screen
C
      IF ( NPE .EQ. 0 ) THEN
         THETA=TR*180./3.14159
         WRITE ( 6 , 200 )IT
  200    FORMAT ( /1X,'NO EXPOSED ELEMENTS FOR THREAT 'I4 )
      END IF
C
C  Check if number of exposed elements is > than the number of elements
C  if so , write error message and stop
C
      IF ( NPE .GT. NELM ) THEN
         WRITE ( 6 , 300 )NPE,NELM
  300    FORMAT ( /1X,'NUMBER OF EXPOSED ELEMENTS = ',I5,/
     .             /1X,'NUMBER OF ELEMENTS = ',I5 )
         STOP
      END IF
C
C  Return
C
      RETURN
C
      END
```

```
C
CCCCC6CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE TRANS
C
CCCCC6CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Trans transforms and projects the c.g. and grid points onto a plane
C  orthogonal to the relative threat vector.
C
C
C  note: for variables in the common block refer to the main listing for
C        defination
C
C  Variable list
C
C    cp = cosine of phi
C    ct = cosine of theta
C    sp = sin of phi
C    st = sin of theta
C    phi = relative threat angle measured from the Earth normal, radians
C    theta = realative threat angle measured from the station velocity
C            vector in the horizontal plane, radians
C    xc = element c.g. x coordinate
C    yc = element c.g. y coordinate
C    zc = element c.g. z coordinate
C    xtc = c.g. x coordinate transformed to orthogonal plane
C    ytc = c.g. y coordinate transformed to orthogonal plane
C    ztc = c.g. z coordinate transformed to orthogonal plane
C    x(j) = x coordinate of jth grid of element i
C    y(j) = y coordinate of jth grid of element i
C    z(j) = z coordinate of jth grid of element i
C
C
C
      INCLUDE 'COMMON1.BLK'
C
C
      DIMENSION X(3),Y(3),Z(3),XT(3),YT(3),ZT(3)
C
C  Get theta and phi from the threat array
C
      THETA=THREAT(1,IT)
      PHI=THREAT(2,IT)
C
C  Calculate the cosine and sin of theta and phi
C
      CT=COS(THETA)
      ST=SIN(THETA)
      CP=COS(PHI)
      SP=SIN(PHI)
C
C  Evaluate the potentially exposed elements
C
      DO 10 I=1,NPE
C
C  Transform coordinates of c.g. of element onto orthogonal plane
C
         XC=ELEMENT( 13,IEXP(1,I) )
         YC=ELEMENT( 14,IEXP(1,I) )
         ZC=ELEMENT( 15,IEXP(1,I) )
C
```

```
         XCT = XC*SP*CT + YC*SP*ST + ZC*CP
         YCT =-XC*ST    + YC*CT
         ZCT =-XC*CP*CT - YC*CP*ST + ZC*SP
C
         TRANSFORM( 2,IEXP(1,I) ) = XCT
         TRANSFORM( 3,IEXP(1,I) ) = YCT
         TRANSFORM( 4,IEXP(1,I) ) = ZCT
C
            DO 20 J=1,3
C
C  Transform grid coordinates onto the orthogonal plane
C
               X(J) = ELEMENT( (1+(J-1)*3) , IEXP(1,I) )
               Y(J) = ELEMENT( (2+(J-1)*3) , IEXP(1,I) )
               Z(J) = ELEMENT( (3+(J-1)*3) , IEXP(1,I) )
C
               XT(J)=  XCT
               YT(J)= -X(J)*ST + Y(J)*CT
               ZT(J)= -X(J)*CP*CT - Y(J)*CP*ST + Z(J)*SP
C
  20        CONTINUE
C
                  DO 30 K=1,3
C
C  Store transformed grid coordinates into transform array
C
                     TRANSFORM((5+(K-1)*3),IEXP(1,I))=XT(K)
                     TRANSFORM((6+(K-1)*3),IEXP(1,I))=YT(K)
                     TRANSFORM((7+(K-1)*3),IEXP(1,I))=ZT(K)
C
C
  30              CONTINUE
C
C
  10 CONTINUE
C
     RETURN
C
     END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE QSORT ( IVAR )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     Sort Reals
C
C     IMPLEMENTATION OF WIRTH'S DOCUMENTED QUICKSORT ALGORITHM.
C
C     GIVEN AN ARRAY OF REAL VALUES AND AN ARRAY OF POINTERS
C     WHICH POINT TO THE VALUES, REARRANGE THE POINTERS SUCH THAT
C     THEY POINT TO VALUES OF INCREASING VALUE
C     BASED ON VALUES IN POINT.
C
C-----INPUT-------------------------------------------------------------
C
C     VALUES         ARRAY OF ELEMENT CG X COORDINATES TO BE SORTED
C     IEXP           ARRAY OF ELEMENT ID'S IN DECENDING ORDER
C     NPE           THE LAST POINTER IS AT POSITION POINT(NPE)
C
C-----OUTPUT------------------------------------------------------------
C
C     IEXP           ARRAY OF ELEMENT ID'S SORTED ACCORDING TO
C                    ORDER OF ARRAY POINT(NPE)
C                    THEY POINT TO INCREASING VALUES OF
C                    CG X COORDINATES OF ELEMENTS
C
C-----VARIABLES---------------------------------------------------------
C
C     TEMP           ARRAY TO TEMPORARILY STORE ARRAY IEXP WHEN SORTING
C                    IEXP BASED ON VALUES IN ARRAY POINT
C     IBEGIN        THE FIRST POINTER IS AT POSITION POINT(IBEGIN)
C     STACK          ARRAY FOR SCRATCH USE ( MUST BE NPE-IBEGIN+1
C                    WORDS LONG)
C
      INCLUDE 'COMMON1.BLK'
C
      INTEGER*2 IBEGIN,IPOINT,II,JJ
      INTEGER*2 POINTER(ISIZE),STACK(2,15),TEMP(ISIZE)
      INTEGER*2 I,J,L,R,S,W
C
      IBEGIN = 1
C
C     INITIALIZE ARRAY POINTER FROM 1 TO NPE
C
      DO 5 I = 1, NPE
    5 POINTER(I)=I
C
CD     WRITE(ALTOUT,10) IBEGIN,NPE,(POINTER(I),I=IBEGIN,NPE)
CD10   FORMAT(' IAC7SI,UNSORTED POINTER,IBEGIN,NPE ',2I5/(' ',10I4))
      S=1
      STACK(1,1)=IBEGIN
      STACK(2,1)=NPE
C
C     TAKE FROM TOP OF STACK
C
  40  CONTINUE
      L=STACK(1,S)
      R=STACK(2,S)
      S=S-1
C
```

```fortran
C     SPLIT  KEY(L).........KEY(R)
C
   80 CONTINUE
      I=L
      J=R
      IPOINT=POINTER((L+R)/2)
C
  120 CONTINUE
      II=POINTER(I)
      IF(TRANSFORM(IVAR,IEXP((IVAR-1),II)).GT.
     .   TRANSFORM(IVAR,IEXP((IVAR-1),IPOINT))) THEN
C
        I=I+1
        GO TO 120
      END IF
C
  160 CONTINUE
      JJ=POINTER(J)
      IF(TRANSFORM(IVAR,IEXP((IVAR-1),IPOINT)).GT.
     .   TRANSFORM(IVAR,IEXP((IVAR-1),JJ))) THEN
C
        J=J-1
        GO TO 160
      END IF
C
      IF(I.LE.J) THEN
        W=POINTER(I)
        POINTER(I)=POINTER(J)
        POINTER(J)=W
        I=I+1
        J=J-1
        IF(I.LE.J) GO TO 120
      END IF
C
      IF(J-L.LT.R-I) THEN
        IF(I.LT.R) THEN
C
C     STACK REQUEST FOR SORTING RIGHT PARTITION
C
          S=S+1
          STACK(1,S)=I
          STACK(2,S)=R
C
        END IF
C
C     CONTINUE SORTING LEFT PARTITION
C
        R=J
C
      ELSE
C
        IF(L.LT.J) THEN
C
C     STACK REQUEST FOR SORTING LEFT PARTITION
C
          S=S+1
          STACK(1,S)=L
          STACK(2,S)=J
        END IF
C
C     CONTINUE SORTING RIGHT PARTITION
C
        L=I
      END IF
C
      IF(L.LT.R) GO TO 80
C
      IF(S.NE.0) GO TO 40
C
CD     WRITE(ALTOUT,500) IBEGIN,NPE,(POINTER(I),I=IBEGIN,NPE)
CD500  FORMAT(' IAC7SI,SORTED POINTER,IBEGIN,NPE ',2I5/(' ',10I4))
C
C     STORE ARRAY IEXP TEMPORARILY IN ARRAY TEMP
C
      DO 180 I = 1 , NPE
  180 TEMP(I) = IEXP((IVAR-1),I)
C
C     SORT ARRAY IEXP BASED ON VALUES IN ARRAY POINTER
C
      DO 200 I = 1 , NPE
C
         INDEX = POINTER(I)
         IEXP((IVAR-1),I) = TEMP(INDEX)
C
  200 CONTINUE
C
      RETURN
C
      END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE SHADOW
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  Shadow determines which elements are exuposed to the current threat.
C  It only considers the elements that were not eliminated by the
C  BACKSIDE subroutine.
C
C  Shadow works with the sorted lists contained in the IEXP array. These
C  lists were sorted on the transformed x & y distances of the element
C  centroids from the threat. An element can be shadowed by another
C  element only if it lies below it in the x sorted list. As a crude
C  test,it's centroid must also lay within the circle enclosing the
C  shadowing element. The origin of this circle is located at the
C  centroid of the shadowing element. This test is accounted for by
C  evaluating the element's position in the y sorted list with respect to
C  the shadowing elements position. If the element passes this test, a
C  precise test is performed to determine if the element's centroid lies
C  within the shadowing element's boundaries. These boundaries have been
C projected onto the plane normal to the threat.
C
C
C  note: for variables in the common block refer to main listing for
C        defination
C
C  Variable list
C
C   direction = logical variable @used to deterrmine which direction to
C               move in the y sorted list
C   el = element number of the potentially shadowed element
C   icr = interger value of rt/srmax
C   icy =   "        "  "  yd/srmax
C   ipx   = position in the x sorted list of the shadowing element
C   ipxel =    "       "  "  "  "     "  "   " pot. shadowed element
C   ipy   =    "       "  "  y  "     "  "   " shadowing element
C   itest = element number of the shadowing element
C   pos = current position in the y sorted list
C   rt = square of the distance from the shadowing elements centroid to
C        the potentially shadowed elements centroid
C   srmax = square of the radius of the circle enclosing the shadowing
C           element with the origin at the centroid
C   yd = square of the y distance from the shadowing elements centroid
C        to the potentially shadowed elements centroid
C   yc = transformed y coordinate of the shadowing elements centroid
C   y1 =     "       "     "      "  "     "           "    grid point 1
C   y2 =     "       "     "      "  "     "           "     "    "   2
C   y3 =     "       "     "      "  "     "           "     "    "   3
C   y4 =     "       "     "      "  " potentially shadowed elements
C        centroid
C   zc = transformed z coordinate of the shadowing elements centroid
C   z1 =     "       "     "      "  "     "           "    grid point 1
C   z2 =     "       "     "      "  "     "           "     "    "   2
C   z3 =     "       "     "      "  "     "           "     "    "   3
C   z4 =     "       "     "      "  " potentially shadowed elements
C        centroid
C
C
C
C
```

```
C
      INCLUDE 'COMMON1.BLK'
C
      INTEGER*2 ICR,ICY,IPX,IPXEL,IPY,ITEST,EL,POS
C
      LOGICAL DIRECTION,INTRI
C
      DIRECTION=.TRUE.
C
C  Starting with the first element in the x sorted list determine if
C  it shades any other elements.
C
      DO 100 I=1,NPE
C
C  Get the shadowing element
C
         ITEST=IEXP(1,I)
C
C
C Get the centroid location and the square of the maximum radius.
C
            YC=TRANSFORM(3,ITEST)
            ZC=TRANSFORM(4,ITEST)
C
            SRMAX=ELEMENT(16,ITEST)
C
C  Determine the shadowing elements location in the x&y sorted list.
C
            IPX=POINT(1,ITEST)
            IPY=POINT(2,ITEST)
C
C  Begin evaluating the elements in the y sorted list , starting at
C  the next element after the shadowing element.
C
            POS=IPY+1
C
C  Determine the potentially shadowed element.
C
   20       EL=IEXP(2,POS)
C
C  Determine the potentially shadowed element's location in the x
C  sorted list.
C
            IPXEL=POINT(1,EL)
C
C  Check if the potentially shadowed element is below the shadowing
C  element in the x sorted list. If not get the next potentially
C  shadowed element.
C
            IF ( IPXEL .GT. IPX ) THEN
C
C  Check if the potentially shadowed element is exposed. If not get the
C  next potentially shadowed element.
C
               IF ( IEL(2,EL) ) THEN
C
C  Since the element is exposed , get it's centroid location.
C
                  Y4=TRANSFORM(3,EL)
                  Z4=TRANSFORM(4,EL)
C
C  Determine the y distance from the shadowing elements centroid to the
C  potentially shadowed elements centroid.
```

```
C
                     YD=(YC-Y4)**2
C
C  Calculate the integer value of the distance to the radius ratio.
C
                     ICY=YD/SRMAX
C
C  Using the integer of the ratio , determine if the element is outside
C  of radius of the shadowing element in the y direction. If so, there is
C  no need to go further through the y sorted list in the present
C  direction. Either switch directions and get next element or get next
C  shadowing element.
C
                     IF ( ICY.NE.0 ) GO TO 40
C
C  Determine the distance from the elements centroid to the shadowing
C  elements centroid and square it.
C

                     RT=YD+(ZC-Z4)**2
C
C  Determine the integer value of the distance to maximum radius ratio.
C
                     ICR=RT/SRMAX
C
C  Use the integer value to determine if the element is inside the radius.
C  If not get the next element.
C
                     IF ( ICR .EQ. 0 ) THEN
C
C  The elements centroid is inside the maximum radius, perform the
C  precision test. First get the shadowing elements grid points location.
C
                        Y1=TRANSFORM(6,ITEST)
                        Z1=TRANSFORM(7,ITEST)
                        Y2=TRANSFORM(9,ITEST)
                        Z2=TRANSFORM(10,ITEST)
                        Y3=TRANSFORM(12,ITEST)
                        Z3=TRANSFORM(13,ITEST)
C
C  Set the exposure logical flag equal to the negative of the logical
C  function test for inside the triangle.
C
                        IEL(2,EL)=.NOT.INTRI(Y1,Z1,Y2,Z2,Y3,Z3,Y4,Z4)
C
                    END IF
C
                 END IF
C
             END IF
C
C  Increment the y list position based on which direction we are
C  currently moving. Check that we do not point to a location outside
C  the list.
C
             IF ( DIRECTION ) THEN
C
                 POS=POS+1
                 IF ( POS.GT.NPE ) GO TO 40
C
             ELSE
C
                 POS=POS-1
```

```
               IF ( POS.LT. 1 ) GO TO 40
C
            END IF
C
C  Get the next potentially shaded element.
C
            GO TO 20
C
C  Change the direction and continue with the new starting positon.
C  Unless we have gone in both directions. Then get next shadowing
C  element.
C
   40       IF ( DIRECTION ) THEN
C
               DIRECTION=.FALSE.
C
               POS=IPY-1
C
               IF ( POS .LT. 1 ) THEN
C
                   DIRECTION=.TRUE.
C
                   GO TO 100
C
               END IF
C
               GO TO 20
C
            ELSE
C
               DIRECTION=.TRUE.
C
            END IF
C
C  Get the next shadowing element.
C
  100 CONTINUE
C
C  Finished
C
      RETURN
C
      END
```

```
C
C
C
      LOGICAL FUNCTION INTRI(AX,AY, BX,BY, CX,CY, PX,PY)
C
C  Return 1 if poinit P is inside triangle or 0 if outside.
C  Triangle has vertices at A,B and C.  P,A,B,C are in the XY plane.
C  EPS is the desired limiting precision in distance units.
C
      INTRI=.FALSE.
      EPS= 1.E-6
C
C  Move origin to A.
C
      PPX= PX - AX
      PPY= PY - AY
      BPX= BX - AX
      BPY= BY - AY
      CPX= CX - AX
      CPY= CY - AY
C
C  Exit if B or C is too near A.
C
      D= SQRT(BPX*BPX + BPY*BPY)
      IF(D .LT. EPS .OR. SQRT(CPX*CPX + CPY*CPY) .LT. EPS) RETURN
C
C  Form base unit vector BU along AB.
C
      BUX= BPX/D
      BUY= BPY/D
C
C  Define F + if ABC is ccw or - if cw, exit if collinear.
C
      F= CPY*BUX - CPX*BUY
      IF(ABS(F).LT.EPS) RETURN
      M= SIGN(1.,F)
C
C  Form altitude unit vector H perpendicular to and on C side of base.
C
      HX= -BUY*M
      HY=  BUX*M
C
C  Heights V and Y of C and P, resp.
C
      V= HX*CPX + HY*CPY
      Y= HX*PPX + HY*PPY
C
C  Outside triangle if P is below base or above C.
C
      IF(Y.LT.0. .OR. Y.GT.V) RETURN
C
C  On-base projections U and X of C and P, resp.
C
      U= BUX*CPX + BUY*CPY
      X= BUX*PPX + BUY*PPY
C
C  P is outside the triangle if X is not between the initersections of
C   the sides with a line through P parallel to the base.
C
      IF(X.LT.U*Y/V .OR. X.GT. D - (D-U)*Y/V) RETURN
C
C  By elimination, the remaining condition is within the triangle.
C
```

```
      INTRI=.TRUE.
C
      END
```

```
C
C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
      SUBROUTINE OUTPUT
C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C  Output writes the pertinent data out to the output file for each
C  element and each relative threat case. The output is in unformatted
C  form.
C
C
C  variable list
C
C    nexp = number of exposed elements
C
C
      INCLUDE 'COMMON1.BLK'
C
      INTEGER*2  I,NEXP
C
C  For the first threat case write out the global threat and element data
C
      IF ( IT.EQ.1 ) THEN
C
C  Write out the analysis type,number of threats, and the total number of
C  elements
C
         WRITE ( 4 ) ITYPE,NT,NELM
C
C  Write out the threat array
C
         DO 10 I=1,NT
            WRITE ( 4 ) (THREAT(J,I),J=1,4)
   10    CONTINUE
C
C  Write out the element id and pid data
C
         DO 20 I=1,NELM
            WRITE ( 4 ) (ID(J,I),J=1,2)
   20    CONTINUE
C
C  Write out the element surface area
C
         DO 30 I=1,NELM
            WRITE ( 4 ) ELEMENT(17,I)
   30    CONTINUE
C
      END IF
C
C  Determine the number of exposed elements
C
      NEXP=0
      DO 40 I=1,NELM
         IF ( IEL(1,I).AND.IEL(2,I) ) THEN
            NEXP=NEXP+1
         END IF
   40 CONTINUE
C
C  Write out the threat case number and the number of exposed elements
C
      WRITE ( 4 ) IT,NEXP
```

```
C
C For each exposed element write out the cosine of the impact angle
C
    DO 50 I=1,NELM
      IF ( IEL(1,I).AND.IEL(2,I) ) THEN
        WRITE ( 4 ) I,TRANSFORM(1,I)
      END IF
  50 CONTINUE
C
C Finished
C
    RETURN
C
C
    END
```

COMMON1.BLK

```
C
C Common block for Geometry Ver 2.5
C
C    ielm = number of elements
C
      PARAMETER (IELM=9000)
C
      INTEGER*2 IT,ITYPE,NELM,NPE,NT,
     1          IEXP(2,IELM),POINT(2,IELM)
C
      INTEGER*4 ID(5,IELM)
C
      REAL*4 ELEMENT(17,IELM),TRANSFORM(13,IELM),THREAT(5,500)
C
      LOGICAL IEL(2,IELM)
C
      COMMON IT,ITYPE,NELM,NPE,NT,
     1       ID,IEL,IEXP,ELEMENT,POINT,THREAT,TRANSFORM
```

APPENDIX C

Source Code for RESPONSE Module

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                     C
C     RESPONSE VER 2.0  5/25/87       C
C                                     C
C        BOEING AEROSPACE CO.         C
C                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Response is a code for producing the hypervelocity impact response
C  file for the BUMPER code. BUMPER calculates the probability of no
C  penetration for spacecraft exposed to meteroids and man-made orbital
C  debris. The  code is limited to the case of aluminum spheres impacting
C  aluminum two plate structures. 30 layers of multi-layer insulation may
C  be included between the plates. The code also considers all impact
C  angles greater than 60 deg ( measured from the normal ) to be equivalent
C  to 60 deg.
C
C  The code was developed under the NASA contract 'Space Station Integrated
C  Wall Design Guide and Penetration Control', by M.A. Wright and
C  A.R. Coronado.
C
C
C
C  Variable list
C
C    aincr = impact angle increment,deg
C    amax = maximum impact angle,deg
C    amin = minimum impact angle,deg
C    ang = impact angle,deg
C    angr = impact angle,radians
C    answer = character string representing user input
C    ctype = configuration type
C                                1- single plate
C                                2 - double plate
C    dia = projectile diameter,in
C    diam =   "          "    ,cm
C    ic = case counter
C    initial = logical variable used to detrmine if current call to
C              diameter is the initial one for the current angle
C    itype = analysis type,1=space debris
C                          2=meteoroids
C    mli = logical variable used to detrmine if 30 layers of mli is
C          included
C    nang = number of angles to be considered
C    nvel = number of velocities to be considerd
C    pfunc = penetration function
C              1-orginal
C              2-pen4
C              3-regression
C    shthk = shield thickness,in
C    stand = shield stand-off,in
C    vele = impact velocity,ft/sec
C    velm =   "        "    ,km/sec
C    vincr = velocity increment,km/sec
C    vmax = maximum velocity,km/sec
C    vmin = minimum velocity,km/sec
C
C
C  Array list
C
C   bhard = array containing the Brinnel hardness values for the
```

```
C           current configuration
C   c - array containing the speed of sound values for the current
C       configuration, ft/sec
C   dens - array containing the density values for the current
C          configuration,lbs/in**3
C   fsu - array containing the shear allowable stress values for the
C         current configuration,psi
C   ftu - array containing the ultimate tensile stress values for the
C         current configuration,psi
C   fy - array containing the yield stress values for the current
C        configuration
C   rtable - array containing the critcal diameters for each case,angle
C            and velocity
C   shpv -array containing the shock projectile velocities for the
C         current configuration
C   wilkc - array containing the values for Wilkinson's constant for
C           the current configuration,km/sec
C
C
C
C
      CHARACTER*20 ANSWER
C
      DIMENSION RTABLE(70,50,10)
C
      DIMENSION BHARD(3),C(3),DENS(3),FSU(3),FTU(3),FY(3),SHPV(3),
     1          WILKC(3)
C
      INTEGER*2 CTYPE,IC,ITYPE,NANG,NVEL,PFUNC
C
      LOGICAL INITIAL,METRIC,MLI
C
C  Initialize variables
C
      IC=0
      PFUNC=0
C
C  Set the angle and velocity limits and increments
C
C  Amin must always =0
C
      AMIN=0.0
C
      AMAX=90.0
      AINCR=5.0
C
C  Determine the number of velocity and angle iterations
C
      NANG = (AMAX-AMIN)/AINCR + 1
C
C  Write header to screen
C
      CALL HEADER (ITYPE)
C
C  Set velocity values based on analysis type
C
C  Vmin must always = 0
C
      VMIN=0.0
C
      IF ( ITYPE.EQ.1 ) THEN
         VMAX=17.0
         VINCR=0.25
      ELSE
         VMAX=70.0
         VINCR=1.0
      END IF
C
         NVEL=(VMAX-VMIN)/VINCR
C
C  Increment the case counter,and determine the wall configuration
C
   10 IC=IC+1
C
      CALL INPUT ( CTYPE,IC,ITYPE,MLI,PFUNC,SHTHK,STAND,VWTHK,BHARD,
     1             C,DENS,FSU,FTU,FY,SHPV,WILKC )
C
C  For the current configuration,determine the critical diameter
C  for each impact angle and velocity
C
      DO 200 I=1,NANG
C
C  Set the angle,in deg and radians
C
         ANG = 0.0 + (I-1)*AINCR
C
C  For angles > 60 deg,set ang=60
C
         IF ( ANG .GT. 60.0 ) ANG=60.0
C
C  Convert ang to radians
C
         ANGR = ANG / 180.0 * 3.141592
C
C  Set initial equal to true
C
         INITIAL=.TRUE.
C
         DO 100 J=1,NVEL
C
C  Set the velocity in ft/sec and km/sec
C
            VELM = J*VINCR
C
C  Convert vel to ft/sec
C
            VELE = VELM * 1.0E+05 / 2.54 / 12.0
C
C  Determine the critical diameter, as a function of wall configuration
C
            IF ( CTYPE.EQ.2 ) THEN
C
C  For the original and regression penetration functions use DOUBLE
C  subroutines
C
               IF ( PFUNC.EQ.1 .OR. PFUNC.EQ.3 ) THEN
                  CALL DOUBLE ( ANGR,DIA,INITIAL,ITYPE,MLI,PFUNC,SHTHK,
     1                          STAND,VELE,VELM,VWTHK,BHARD,C,DENS,
     2                          FSU,FTU,FY,SHPV,WILKC )
C
C  For Pen4 use the pen4 subroutine
C
               ELSE IF ( PFUNC.EQ.2 ) THEN
                  CALL PEN4 ( ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                        INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
               END IF
```

```
C
            ELSE
               CALL SINGLE ( ANGR,DIA,VELE,VWTHK,DENS,FTU )
            END IF
C
C  Convert the diameter to cm
C
            DIAM = DIA * 2.54
C
C  Store the diameter in RTABLE
C
            RTABLE(J,I,IC)=DIAM
C
  100    CONTINUE
C
  200 CONTINUE
C
C  Determine if another case is to be run
C
      WRITE ( 6,300 )
  300 FORMAT(/1X,'DO YOU WISH TO RUN ANOTHER CASE ? <CR>=YES : ',$)
      READ ( 5,310 ) ANSWER
  310 FORMAT ( A )
C
      IF ( ANSWER(1:1).NE.'N' ) GO TO 10
C
C  Write response information to output file
C
      WRITE ( 8 )ITYPE,IC
C
      WRITE ( 8 ) NANG,AINCR
C
      WRITE ( 8 ) NVEL,VINCR
C
      DO 500 I=1,IC
         DO 450 J=1,NANG
            DO 400 K=1,NVEL
               WRITE ( 8 ) RTABLE(K,J,I)
  400       CONTINUE
  450    CONTINUE
  500 CONTINUE
C
C  Close the output files
C
      CLOSE ( UNIT=7,STATUS='KEEP')
      CLOSE ( UNIT=8,STATUS='KEEP')
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE HEADER (ITYPE)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C  Header writes the header to the screen and determines the analysis
C  type.
C
C
C
      INTEGER*2 ITYPE
C
      CHARACTER*80 ANSWER
C
      WRITE ( 6,10 )
   10 FORMAT ( /1X,'*********************',//,2X,'RESPONSE VER 2.0',
     1          //,1X,'Last Update 5/25/87',//,'*********************')
.C
C  Determine the analysis type
C
   20 WRITE ( 6,30 )
   30 FORMAT ( /1X,'ANALYSIS TYPE ?',/,2X,'1-DEBRIS <CR>',/,2X,
     1          '2-METEOROIDS',/,1X,'ANSWER 1 OR 2 : ',$)
C
      READ ( 5,40 ) ANSWER
   40 FORMAT (A)
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         ITYPE=1
      ELSE
         READ ( ANSWER(1:80),50 ) ITYPE
   50    FORMAT ( BN,I4 )
      END IF
C
      IF ( ITYPE.EQ.1 .OR. ITYPE.EQ.2 ) THEN
         CONTINUE
      ELSE
         WRITE ( 6,60 )
   60    FORMAT ( /1X,'INCORRECT INPUT' )
         GO TO 20
      END IF
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE INPUT (CTYPE,IC,ITYPE,MLI,PFUNC,SHTHK,STAND,VWTHK,
     1                  BHARD,C,DENS,FSU,FTU,FY,SHPV,WILKC )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Input reads in the physical properties file, determines the wall
C  configuration, and writes the configuration information out to the
C  properties output file.
C
C
C  Variable list
C
C   answer = character string representing user input
C   ctype = configuration type
C                                 1- single plate
C                                 2- double plate
C   ds = epson printer control for double space
C   ic = current case number
C   itype = analysis type
C   lunits = length unit,cm for metric, in for english
C   metric = a logical variable used to determine if the input is in
C            metric units
C   mli = logical variable used to determine if multi-layer insulation
C         is used between the shield and vessel wall
C   ml = length of the material menu
C   nc = material counter < 10
C   ofile1 = properties output file
C   ofile2 = response output file
C   pfunc = penetration function
C            1- orginal
C            2- pen4
C            3- regression
C   pid = property id
C   shthk = shield thickness,in or cm
C   stand = shield stand-off distance,in or cm
C   tp = epson printer control for top of page
C   vwthk = vessel wall thickness,in or cm
C
C  Array list
C
C   bhard = array containing the shield ,vessel wall, and projectile
C           Brinnel hardness
C   c = array containing the shield, vessel wall, and projectile speed
C       of sound in ft/sec
C   constant = array containing the values of Wilkinson's constant for all
C              materials in the physical properties file, km/sec
C   dens = array containing the shield, vessel wall, and projectile density
C          in lbs/in**3
C   density = array containing the value of the density for all the materials
C             in lb/in**3
C   hard = array containing the values of the Brinnel hardness for all the
C          materials
C   fsu = array containing the shield, vessel wall, and projectile shear
C         stress, psi
C   ftu = array containing the shield, vessel wall, and projectile ultimate
C         tensile stress, psi
C   fty = array containing the shield, vessel wall, and projectile tensile
C         yield stress, psi
C   mat = array containing the material number for the shield, vessel wall,
C          and the projectile respectively
C   material - array containing all of the material names for all of the
C                 materials in the physical properties file
C   shear - array containing the values of the shear stress for all of the
C             materials, psi
C   shock - array containing the values of the shock projectile velocity
C             for all of the materials, values obtained from Physics
C   shpv - array containing the shield, vessel wall, all projectile shock
C            projectile velocities
C   sound - array containing the values of the speed of sound for all of
C             the materials,ft/sec
C   wilkc - array containing the shield, vessel wall, and projectile
C             Wilkkinson's constant,km/sec
C   yield - array containing the values of the yield stress for all the
C             materials,psi
C   ult - array containing the values of the ultimate tensile stress for
C           all of the materials,psi
C
C
C
      CHARACTER*1 DS,TP
      CHARACTER*2 LUNITS
      CHARACTER*12 MATERIAL(10)
C
      CHARACTER*80 ANSWER,OFILE1,OFILE2
C
      DIMENSION BHARD(3),C(3),DENS(3),FSU(3),FTU(3),FY(3),SHPV(3),
     .          WILKC(3)
C
      DIMENSION DENSITY(10),CONSTANT(10),HARD(10),SHEAR(10),SHOCK(10),
     .          SOUND(10),ULT(10),YIELD(10)
C
      INTEGER*2 CTYPE,IC,ITYPE,ML,NC,PFUNC,PID,MAT(3)
C
      LOGICAL METRIC,MLI
C
      SAVE
C
C For the first case only, initialize the variables & read in the
C material properties file.
C
      IF (IC.GT.1) GO TO 70
C
      NC=1
      METRIC=.TRUE.
      MLI=.FALSE.
C
      OPEN (UNIT=2,FILE='MAT.PRP',STATUS='OLD')
C
   10 READ ( 2,20,END=30 )MATERIAL(NC),DENSITY(NC),YIELD(NC),
     1                     ULT(NC),SHEAR(NC),CONSTANT(NC),SOUND(NC),
     2                     SHOCK(NC),HARD(NC)
   20 FORMAT (A,BN,8E12.5)
C
      ML=NC
      NC=NC+1
C
      GO TO 10
C
   30 CLOSE ( UNIT=2,STATUS='KEEP' )
C
C Read in the output file names.
C
```

```
      WRITE ( 6,40 )
   40 FORMAT (/1X,'OUTPUT FILENAME FOR WALL PROPERTIES ',
     .        ' <CR>=RESPONSE.SUM : ',$)
      READ ( 5,'(A)' )ANSWER
      IF ( ANSWER (1:1) .EQ. ' ' ) ANSWER='RESPONSE.SUM'
      OFILE1=ANSWER
C
      WRITE ( 6,50 )
   50 FORMAT (/1X,'OUTPUT FILENAME FOR RESPONSE TABLES ',
     .         '<CR>=STATION.RSP : ',$)
      READ ( 5,'(A)' )ANSWER
      IF ( ANSWER(1:1).EQ.' ') ANSWER='STATION.RSP'
      OFILE2=ANSWER
C
C  Open the output files.
C
      OPEN ( UNIT=7, FILE=OFILE1, STATUS='UNKNOWN' )
      OPEN ( UNIT=8, FILE=OFILE2, STATUS='UNKNOWN',FORM='UNFORMATTED' )
C
      REWIND 7
      REWIND 8
C
C  Determine the type of units for input.
C
      WRITE ( 6,60 )
   60 FORMAT (/1X,'INPUT IN METRIC OR ENGLISH UNITS  <CR>=METRIC : ',$)
      READ ( 5,'(A)' )ANSWER
      IF ( ANSWER(1:1).EQ.'E' ) METRIC=.FALSE.
C
C  Set the units and if they are metric convert the variables to metric.
C
   70 IF ( METRIC ) THEN
        LUNITS='CM'
        SHTHK=SHTHK*2.54
        VWTHK=VWTHK*2.54
        STAND=STAND*2.54
      ELSE
        LUNITS='IN'
      END IF
C
C  Set the property id to the current case number and display
C
      PID=IC
C
      WRITE ( 6,80 )IC
   80 FORMAT (/1X,'PROPERTY ID NUMBER = ',I2 )
C
C  Determine configuration type
C
   90 WRITE ( 6,100 )
  100 FORMAT ( /1X,'CONFIGURATION TYPE',/,5X,'1- SINGLE PLATE ',/,
     1         5X,'2- DOUBLE PLATE <CR>',/,1X,'ANSWER (1 or 2) : ',$)
      READ ( 5,'(A)' ) ANSWER
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         CTYPE=2
      ELSE
         READ ( ANSWER(1:12),110,ERR=90 ) CTYPE
  110    FORMAT ( BN,I4 )
      END IF
C
C  Check that the input was correct
C
      IF ( CTYPE.LT.1 .OR. CTYPE.GT.2 ) THEN
```

```
         WRITE ( 6,120 )
  120    FORMAT( /1X,'INCORRECT INPUT' )
         GO TO 90
      END IF
C
C  For single plate configuration skip down to the vessel wall material
C
      IF ( CTYPE.EQ.1 ) GO TO 290
C
C  Determine which double wall penetration function to use for the first
C  double wall case only
C
      IF ( PFUNC.GT.0 )  GO TO 150
C
  130 WRITE ( 6,140 )
  140 FORMAT ( /1X,'PENETRATION FUNCTION ',/,5X,'1-ORIGINAL <CR>',/,
     1         5X,'2-PEN4',/,5X,'3-REGRESSION',/,1X,'ANSWER (1-3) : ',$)
      READ ( 5,'(A)' ) ANSWER
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         PFUNC=1
      ELSE
         READ ( ANSWER(1:80),'(BN,I4)',ERR=130 ) PFUNC
      END IF
C
C  Check Input
C
      IF ( PFUNC.LT.1 .OR. PFUNC.GT.3 ) GO TO 130
C
C  Determine the shield material.
C
  150 WRITE ( 6,160 )
  160 FORMAT (/1X,'SHIELD MATERIAL ')
C
C  Write out the material list.
C
      DO 180 I=1,ML
C
         WRITE ( 6,170 )I,MATERIAL(I)
  170    FORMAT ( 3X,I2,'- ',A )
C
  180 CONTINUE
C
C  For the initial case, set the material default number equal to one.
C  For all other cases use the previous shield material number as the
C  default. If an error is detected on the read, repeat the process.
C
      IF ( IC .EQ. 1 ) THEN
  190    WRITE ( 6,220 )
         READ ( 5,'(A)' ) ANSWER
         IF ( ANSWER(1:1).EQ.' ')ANSWER='1'
         READ(ANSWER(1:4),200,ERR=190)MAT(1)
  200    FORMAT(BN,I4)
      ELSE
  210    WRITE ( 6,230 ) MAT(1)
         READ ( 5,'(A)') ANSWER
         IF (ANSWER(1:1).NE.' ') THEN
            READ ( ANSWER(1:4),200,ERR=210 ) MAT(1)
         END IF
      ENDIF
  220 FORMAT (1X,'SELECT MATERIAL NUMBER <CR>=1 : ',$)
  230 FORMAT (1X,'SELECT MATERIAL NUMBER <CR>=',I2,' : ',$)
C
C  Check that the value read in is  contained in the list.
```

```
C
      IF ( MAT(1) .LT.1 .OR. MAT(1).GT. ML ) GO TO 150
C
C  Determine the shield thickness. For the initial case there is no default,
C  for all other cases use the previous value as the default.
C
      IF ( IC.EQ.1 ) THEN
  240    WRITE ( 6,270 )LUNITS
         READ ( 5,*,ERR=240 ) SHTHK
      ELSE
  250    WRITE ( 6,280 ) LUNITS,SHTHK
         READ ( 5,'(A)' ) ANSWER
         IF ( ANSWER(1:1).NE.' ') THEN
            READ ( ANSWER(1:12),260,ERR=250 ) SHTHK
  260       FORMAT(BN,E12.0)
         END IF
      END IF
  270 FORMAT ( /1X,'SHIELD THICKNESS (',A,') = : ',$)
  280 FORMAT ( /1X,'SHIELD THICKNESS (',A,') = ',F8.3,' : ',$)
C
C  Determine the vessel wall material. Use the same technique as used
C  to determine the shield material.
C
  290 WRITE ( 6,300 )
  300 FORMAT (/1X,'VESSEL WALL MATERIAL ' )
C
      DO 310 I=1,ML
         WRITE ( 6,170 )I,MATERIAL(I)
  310 CONTINUE
C
      IF ( IC.EQ.1 ) THEN
  320    WRITE ( 6 ,220 )
         READ ( 5,'(A)'  ) ANSWER
         IF ( ANSWER(1:1) .EQ. ' ' ) ANSWER='1'
         READ ( ANSWER (1:4),200,ERR=320 ) MAT(2)
      ELSE
  330    WRITE ( 6,230 ) MAT(2)
         READ ( 5,'(A)' ) ANSWER
         IF ( ANSWER(1:1).NE.' ') THEN
            READ ( ANSWER(1:4),200,ERR=330 )MAT(2)
         END IF
      END IF
C
      IF ( MAT(2).LT.1 .OR. MAT(2).GT.ML ) GO TO 290
C
C  Determine the vessel wall thickness.
C
      IF ( IC .EQ. 1 ) THEN
  340    WRITE ( 6,360 )LUNITS
         READ ( 5,*,ERR=340 ) VWTHK
      ELSE
  350    WRITE ( 6,370 ) LUNITS,VWTHK
         READ ( 5,'(A)' ) ANSWER
         IF ( ANSWER(1:1).NE.' ') THEN
            READ (ANSWER(1:12),260,ERR=350) VWTHK
         END IF
      END IF
  360 FORMAT (/1X,'VESSEL WALL THICKNESS (',A,') = : ',$)
  370 FORMAT (/1X,'VESSEL WALL THICKNESS (',A,') = ',F8.3,' : ',$)
C
C  For single plate configuration skip stand-off and mli
C
      IF ( CTYPE.EQ.1 ) GO TO 430
```

```
C
C  Determine the shield stand-off distance.
C
       IF ( IC.EQ.1 ) THEN
 380      WRITE ( 6,400 )LUNITS
          READ ( 5,*,ERR=380 ) STAND
       ELSE
 390      WRITE ( 6,410 )LUNITS,STAND
          READ ( 5,'(A)' ) ANSWER
          IF ( ANSWER(1:1).NE.' ') THEN
             READ ( ANSWER(1:12),260,ERR=390 ) STAND
          END IF
       END IF
 400   FORMAT ( /1X,'SHIELD STAND-OFF (',A,') = : ',$)
 410   FORMAT ( /1X,'SHIELD STAND-OFF (',A,') = ',F8.3,' : ',$)
C
C  Determine if MLI is to be included, but not for the pen4 penetration
C  function
C
       IF ( PFUNC.EQ.2 ) GO TO 430
C
       WRITE ( 6,420 )
 420   FORMAT (/1X,'INCLUDE 30 LAYERS OF MLI BETWEEN PLATES <CR>=YES : ',
     1         $)
       READ ( 5,'(A)' )ANSWER
       IF ( ANSWER(1:1).EQ.' ' .OR. ANSWER(1:1).EQ.'Y' )MLI=.TRUE.
C
C  Set the projectile material property based on analysis type
C  For debris use 100-0 al, for meteoroids only density is important use
C  0.50 g/cc .
C
 430   IF ( ITYPE.EQ.1 ) THEN
          BHARD(3)=23.0
          C(3)=16550.0
          DENS(3)=0.098
          FSU(3)=5000.0
          FTU(3)=13000.0
          FY(3)=9000.0
          SHPV(3)=1.345
          WILKC(3)=.126
       ELSE
          BHARD(3)=0.0
          C(3)=0.0
          DENS(3)=0.50/27.705
          FSU(3)=0.0
          FTU(3)=0.0
          FY(3)=0.0
          SHPV(3)=0.0
          WILKC(3)=0.0
       END IF

C
C  Write out the inputed values to the properties output file.
C
       TP=CHAR(12)
       DS=CHAR(10)
C
       WRITE ( 7,440 )TP
 440   FORMAT ( A,'RESPONSE VER 2.0 ')
C
       IF ( ITYPE.EQ.1 ) THEN
          WRITE ( 7,450 ) DS
       ELSE
```

```
         WRITE ( 7,460 ) DS
      END IF
  450 FORMAT (A,'MAN-MADE ORBITAL DEBRIS ANALYSIS' )
  460 FORMAT (A,'METEOROID ANALYSIS ' )
C
      WRITE ( 7,465 ) DS,PID
  465 FORMAT(A,'PROPERTY ID ',I4)
C
      IF ( CTYPE.EQ.1 ) THEN
         WRITE ( 7,470 )DS
      ELSE
         WRITE ( 7,480 )DS
      END IF
  470 FORMAT ( A,'SINGLE PLATE' )
  480 FORMAT ( A,'DOUBLE PLATE' )
C
      IF ( CTYPE.EQ.1 ) GO TO 520
C
      IF ( PFUNC.EQ.1 ) THEN
         WRITE ( 7,485) DS
      ELSE IF ( PFUNC.EQ.2 ) THEN
         WRITE ( 7,486) DS
      ELSE IF ( PFUNC.EQ.3 ) THEN
         WRITE ( 7,487) DS
      END IF
  485 FORMAT ( A,'ORIGINAL PENETRATION FUNCTION' )
  486 FORMAT ( A,'PEN4 PENETRATION FUNCTION' )
  487 FORMAT ( A,'REGRESSION PENETRATION FUNCTION' )
C
      WRITE ( 7,500 )DS,MATERIAL(MAT(1))
  500 FORMAT(A,'SHIELD MATERIAL = ',A)
C
      WRITE ( 7,510 )DS,LUNITS,SHTHK
  510 FORMAT(A,'SHIELD THICKNESS (',A,') = ',F8.4)
C
  520 WRITE ( 7,530 )DS,MATERIAL(MAT(2))
  530 FORMAT(A,'VESSEL WALL MATERIAL = ',A)
C
      WRITE ( 7,540 )DS,LUNITS,VWTHK
  540 FORMAT(A,'VESSEL WALL THICKNESS (',A,') = ',F8.4)
C
      IF ( CTYPE.EQ.1 ) GO TO 570
C
      WRITE ( 7,550 )DS,LUNITS,STAND
  550 FORMAT(A,'SHIELD STAND-OFF (',A,') = ',F8.4)
C
      IF ( MLI ) THEN
         WRITE ( 7,560 )DS
  560    FORMAT(A,'30 LAYERS OF MLI BETWEEN SHIELD AND VESSEL WALL')
      END IF
C
C  If the variables were read in in metric units, convert to english.
C
  570 IF ( METRIC ) THEN
         SHTHK=SHTHK/2.54
         VWTHK=VWTHK/2.54
         STAND=STAND/2.54
      END IF
C
C  Build the material properties arrays.
C
      DO 580 I=1,2
C
```

```
         BHARD(I)=HARD(MAT(I))
         C(I)=SOUND(MAT(I))
         DENS(I)=DENSITY(MAT(I))
         FSU(I)=SHEAR(MAT(I))
         FTU(I)=ULT(MAT(I))
         FY(I)=YIELD(MAT(I))
         SHPV(I)=SHOCK(MAT(I))
         WILKC(I)=CONSTANT(MAT(I))
C
  580 CONTINUE
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE DOUBLE (ANGR,DIA,INITIAL,ITYPE,MLI,PFUNC,SHTHK,STAND,
     .                   VELE,VELM,VWTHK,BHARD,C,DENS,FSU,FTU,FY,
     .                   SHPV,WILKC)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  DOUBLE DETERMINES THE PROJECTILE DIAMETER THAT JUST PENETRATES
C  THE GIVEN DOUBLE PLATE CONFIGURATION AT THE GIVEN IMPACT VELOCITY
C  AND ANGLE. IT IS USED FOR BOTH THE ORGINAL AND REGRESSION
C  PENETRATION FUNCTIONS.
C
C
C
C     VARIABLE LIST
C
C     ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL,RADIANS
C     BALL = LOGICAL PARAMETER USED TO DETERMINE IF THE BALLITIC
C            SUBROUTINES ARE CALLED
C     BHARD = ARRAY CONTAINING THE VALUES OF THE BRINNEL HARDNESS FOR
C             THE SHIELD & VESSEL WALL MATERIALS
C     BRIST = LOGICAL PARAMETER USED TO DETERMINE IF THE FRAGMENTING
C             SUBROUTINES ARE CALLED
C     C = ARRAY CONTAINING VALUES FOR THE SPEED OF SOUND (FT/SEC) FOR
C         THE SHIELD AND VESSEL WALL MATERIALS
C     DIA = PROJECTILE DIAMETER  ( IN. )
C     DIAB = DIAMETER AS DETERMINED BY SUBROUTINE BRISTOW ( IN. )
C     DIABL = DIAMETER AS DETERMINED BY SUBROUTINE BALLIST ( IN. )
C     DIAW = DIAMETER AS DETERMINED BY SUBROUTINE WILKIN  ( IN. )
C     EVWTHK = EQUIVALENT VESSEL WALL THK.  ( IN. )
C     FY = ARRAY CONTAINING THE VALUES OF THE YIELD STRENGTH FOR THE
C          SHIELD & VESSEL WALL MATERIALS
C     FSU = ARRAY CONTAING VALUES OF THE ULTIMATE SHEAR STRENGTH (PSI)
C           FOR THE SHIELD AND VESSEL WALL MATERIALS
C     FTU = ARRAY CONTAING VALUES OF THE ULTIMATE TENSILE STRENGTH (PSI)
C           FOR THE SHIELD AND VESSEL WALL MATERIALS
C     INITIAL = LOGICAL PARAMETER USED TO DETERMINE IF CURRENT
C               SUBROUTINE CALL IS INITIAL ONE
C     ITYPE = ANALYSIS TYPE 1=DEBRIS & METEOROIDS, 2=METEOROIDS
C     MLI = CHARACTER STRING USED TO DETERMINE IF MLI IS USED IN WALL
C     PFUNC = PENETRATION FUNCTION
C                1- ORIGINAL
C                2- PEN4
C                3- REGRESSION
C     SHATTER = LOGICAL PARAMETER USED TO DETERMINE IF PEN4 BALLISTIC
C                LIMITS HAVE BEEN EXCEEDED
C     SHPV = ARRAY CONTAINING THE VALUES FOR THE SHOCK PROJECTILE
C            VELOCITY (FT/SEC ) OF THE SHIELD & VESSEL WALL MATERIALS
C     SHTHK = SHIELD THICKNESS ( IN. )
C     STAND = SHIELD STAND-OFF DISTANCE (IN.)
C     THKMLI = EQUIVALENT THK OF 30 LAYERS OF MLI  ( IN. )
C     VELE = VEL IN FT/SEC
C     VELM = VEL IN KM/SEC
C     VWTHK = VESSEL WALL THICKNESS ( IN. )
C     WILKC = ARRAY  CONTAINING THE VALUES OF WILKINSON'S CONSTANT FOR
C             THE SHIELD & VESSEL WALL MATERIALS
C
      INTEGER*2 ITYPE,PFUNC
C
      DIMENSION  BHARD(3),C(3),DENS(3),FY(3),FSU(3),FTU(3),SHPV(3),
     .           WILKC(3)
C
      LOGICAL BALL,BRIST,INITIAL,MLI,SHATTER
C
C  INITIALIZE VARIABLES
C
      IF ( INITIAL ) THEN
         BALL=.TRUE.
         BRIST=.TRUE.
         DIA=.01
         SHATTER=.FALSE.
      END IF
C
C  TAKE 30 LAYERS OF MLI INTO ACCOUNT USING COUR-PALAIS'S EQN.
C  SET LIMIT FOR EQN. AT 10 KM/SEC  OR SKIP OVER SECTION ENTIRELY
C
C
      IF (.NOT. MLI ) THEN
         THKMLI = 0.0
         GO TO 50
      ENDIF
C
      IF ( VELM .LT. 10.0 ) THEN
         THKMLI = 3.045E-06 * ( VELM ** 3.42 )
      ELSE
         THKMLI = 3.045E-06 * ( 10.0 ** 3.42 )
      END IF
C
C  CONVERT TO IN.
C
      THKMLI = THKMLI / 2.54
C
C  ADD EQUIVALENT THK. OF MLI TO VESSEL WALL THK.
C
50    EVWTHK = VWTHK + THKMLI
C
C
C  DETERMINE DIAMETER IN IN. THAT  PENETRATES THE SHIELD AND
C  JUST DOES NOT PENETRATE THE VESSEL WALL
C
C  IF THE ANALYSIS IS FOR METEOROIDS, THEN ONLY USE WILKINSON'S METHOD
C  TO DETERMINE THE PENETRATION DIAMETER
C
C
      IF ( ITYPE .EQ. 2 .OR. VELM .GT. 12.0 )  THEN
          BALL=.FALSE.
          BRIST=.FALSE.
          GOTO 500
      ENDIF
C
C  INITIALLY CALCULATE THE PENETRATION DIAMETER USING BOTH THE
C  BALLISTIC AND FRAGMENTING SUBROUTINES. THE DIAMETER CALCULATED BY
C  THE BALLISTIC SUBROUTINE IS USED UNTIL THE VALUE CALCULATED FROM
C  THE FRAGMENTING SUBROUTINE IS GREATER. AT THAT TIME IT IS NO LONGER
C  NECESSARY TO CALL THE BALLISTIC SUBROUTINES.
C
C  FOR THE ORGINAL PENETRATION FUNCTION THE OLD VERSION OF PEN4 IS USED
C  IN THE BALLISTIC REGIME AND THE BURCH MODIFIED EQUATIONS ARE USED
C  IN THE FRAGMENTING REGIME.
C
C  FOR THE REGRESSION PENETRATION FUNCTION THE NEW PEN4 IS USED IN
C  THE BALLISTIC REGIME AND THE REGRESSION EQUATIONS ARE USED IN
C  FRAGMENTING REGIME.
```

```
C
C
      IF(BALL)THEN
        IF ( PFUNC.EQ.1 ) THEN
           CALL BALLIST (ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                   INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE)
        ELSE
           CALL PEN4 (  ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                  INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
        END IF
           DIABL=DIA
C
        IF ( PFUNC.EQ.1. ) THEN
           CALL BRISTOW (ANGR,C,DIA,EVWTHK,INITIAL,PFUNC,SHTHK,STAND,
     1                   VELE )
        ELSE
           CALL REGRESS (  ANGR,DIA,MLI,SHTHK,STAND,VELM,VWTHK )
        END IF
        DIAB=DIA
C
C  CHECK IF THE DIAMETER CALCULATED BY BALLISTIC SUBROUTINE IS LESS
C  THAN THAT CALCULATED BY FRAGMENTING SUBROUTINE. IF SO SET BALL TO
C  FALSE .
C
        IF(DIAB.GT.DIABL)THEN
          BALL=.FALSE.
          DIA=DIAB
          GOTO 700
        ELSE
          BALL=.TRUE.
          DIA=DIABL
          GOTO 700
        ENDIF
C
C  CALCULATE THE PENETRATION DIAMETER USING BOTH THE FRAGMENTING AND
C  WILKIN SUBROUTINES. THE DIAMETR CALCULATED BY FRAGMENTING IS USED
C  UNTIL THE VALUE DETERMINED BY WILKIN IS LESS. IT IS THEN NOT
C  NECESSARY TO CALL FRAGMENTING.
C
C
      ELSE
        IF(BRIST)THEN
          IF ( PFUNC.EQ.1. ) THEN
             CALL BRISTOW (ANGR,C,DIA,EVWTHK,INITIAL,PFUNC,SHTHK,STAND,
     1                     VELE )
           ELSE
              CALL REGRESS (  ANGR,DIA,MLI,SHTHK,STAND,VELM,VWTHK )
           END IF
           DIAB=DIA
C
           CALL WILKIN (ANGR,DENS,DIA,EVWTHK,ITYPE,SHTHK,STAND,VELM,
     .                  WILKC)
           DIAW=DIA
C
C  CHECK IF THE VALUE DETERMINED BY WILKIN IS LESS THEN THAT
C  DETERMINED BY BRISTOW. IF SO SET BRIST TO FALSE.
C
           IF(DIAW.LT.DIAB)THEN
             BRIST=.FALSE.
             DIA=DIAW
             GOTO 700
           ELSE
             BRIST=.TRUE.
```

```
               DIA=DIAB
               GOTO 700
            ENDIF
C
         ENDIF
C
      ENDIF
C
C
C  CALCULATE THE DIAMETER USING THE WILKIN SUBROUTINE.
C
500   CALL WILKIN (ANGR,DENS,DIA,EVWTHK,ITYPE,SHTHK,STAND,VELM,WILKC)
C
C  SET INITIAL TO FALSE
C
700   INITIAL = .FALSE.
C
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE BRISTOW  (ANGR,C,DIA,EVWTHK,INITIAL,SHTHK,STAND,
     1                     VELE )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C     BRISTOW DETERMINES NUMBER OF PLATES PENETRATED. BASED ON
C     "MULTIPLE-DAMAGE STUDY", BY G.T. BURCH, BOEING 1967,AIR FORCE
C     ARMAMENT LABORATORY TECH REPORT AFATL-TR-67-116.
C
C     VALID FROM 3.6 TO 10.2 KM/SEC,AND FOR ALUMINUM PROJECTILES
C     IMPACTING ALUMINUM PLATES AT NORMAL .
C
C
C     VARIABLE LIST
C
C       ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL, RADIANS
C       C1 = SPEED OF SOUND IN SHIELD MATERIAL,FT/SEC
C       C = ARRAY CONTAINING VALUES OF THE SPEED OF SOUND FOR
C           THE SHIELD AND VESSEL WALL MATERIALS ( FT/SEC )
C       CHI = INTERMEDIATE VARIABLE
C       DIA = PROJECTILE DIAMETER,IN.
C       DIA1 = PREVIOUS DIAMETER
C       DIA2 = CURRENT DIAMETER
C       DT = INTERMEDIATE VARIABLE
C       EVWTHK = EQUIVALENT VESSEL WALL THICKNESS,IN.
C       F1 = INTERMEDIATE VARIABLE
C       F2 =      "           "
C       F3 =      "           "
C       I = ITERATION COUNTER
C       INITIAL = LOGICAL PARAMETER USED TO DETERMINE IF CURRENT CALL
C                IS INITIAL ONE
C       NF = NUMBER OF PLATES PENETRATED BY THE FLIGHT PATH COMPONENT
C       NN = NUMBER OF PLATES PENETRATED BY THE NORMAL COMPONENT
C       PLP = # OF PLATES PENETRATED
C       PLP1 = PREVIOUS PLP
C       PLP2 = CURRENT PLP
C       SD = INTERMEDIATE VARIABLE
C       SHTHK = SHIELD THICKNESS,IN.
C       SIN3 = INTERMEDIATE VARIABLE
C       SLOPE = SLOPE OF LINE THROUGH (DIA1,PLP1) & (DIA2,PLP2)
C       SPF = NUMBER OF PLATES PENETRATED THAT ACCOUNTS FOR SPALLING
C       STAND = SHIELD STAND-OFF (IN.)
C       SWITCH = LOGICAL VARABLE USED TO DETERMINE IF THE CRITICAL
C                DIAMETER IS TO BE ESTIMATED THROUGH LINEAR
C                INTERPOLATION
C       T1D = INTERMEDIATE VARIABLE
C       T2D =         "
C       TEST = RATIO OF THE NUMBER OF PLATES PENETRATED TO THE
C              SPALLING FACTOR
C       VC =          "
C       VELE = VEL IN FT/SEC
C       VHOLD = VELOCITY HOLDER
C
C
C
      DIMENSION C(3)
C
      LOGICAL INITIAL
```

```
C
      REAL NF,NN
C
      LOGICAL SWITCH
C
      SAVE PLP1,PLP2,DIA1,DIA2
C
C
C  SINCE THIS METHOD DOES NOT SOLVE FOR THE DIAMETER THAT JUST
C  PENETRATES DIRECTLY AN ITERATIVE APPROACH IS TAKEN. INITIAL VALUES
C  ARE SET,AND USED TO APPROXIMATE THE CORRECT VALUE. THIS PROCESS
C  IS CONTINUED UNTIL THE ANSWER IS WITHIN TOLERANCES.
C
C
C  SET INITIAL VALUES
C
      I=0
      SPF = 0.85
      SWITCH=.FALSE.
C
C  IF THIS IS THE INITIAL CALL SET INITIAL  ELSE USE PREVIOUS VALUES
C  VALUES
C
      IF (INITIAL) THEN
         DIA2=10.0
      ENDIF
C
C   FOR VELOCITIES LESS THAN 11,800 FT/SEC SET THE VELOCITY EQUAL
C   TO 11,800 AND CALCULATE THE CRITICAL DIAMETER. USE THIS VALUE
C   TO ESTIMATE THE ACTUAL DIAMETER. SET THE LOGICAL VARIABLE SWITCH
C   TO TRUE AND SAVE THE VELOCITY AS VHOLD.
C
      IF ( VELE .LT. 11800. ) THEN
         VHOLD=VELE
         VELE=11800.
         SWITCH=.TRUE.
      END IF
C
C  CALCULATE INTERMEDIATE VARIABLES THAT DO NOT NEED TO BE CALCULATED
C  FOR EACH DIAMETER
C
      CHI=TAN(ANGR)-0.50
      SIN3=(SIN(ANGR))**3
      VC=VELE/C(1)
C
100   I=I+1
C
C  IF THIS IS THE FIRST PREDICTION USE THE DIAMETER VALUE THAT WAS
C  CALCULATED TO PENETRATE FOR THE PREVIOUS CASE AS A STARTING POINT.
C  ELSE USE A LINEAR PREDICTION APPROACH BASED  ON THE LAST TWO
C  PREDICTIONS.
C
      IF ( I.EQ.1 ) THEN
C
         DIA=DIA2
         DIA2=0.0
         PLP2=0.0
C
      ELSE
C
         SLOPE=(PLP2-PLP1)/(DIA2-DIA1)
         DIA=((SPF-PLP1)/SLOPE+DIA1)
C
```

```
C  CHECK THAT DIA > 0.0
C
         IF(DIA.LT.0.0) DIA=1.0E-10
C
C  Check if dia1 = dia2,if so stop
C
         IF ( DIA.EQ.DIA2 ) THEN
            WRITE ( 6,150 ) ANGR,VELE,DIA
  150       FORMAT (/1X,'BRISTOW CANNOT CONVERGE BECAUSE OF FLAT ',
     1              'SLOPE (ANGLE,VEL,DIA) =',3E12.5 )
            STOP
         END IF
      END IF
C
C  CALCULATE # OF PLATES PENETRATED
C
      SD=STAND/DIA
      T1D=SHTHK/DIA
      T2D=EVWTHK/DIA
C
      F1=2.42*(T1D**(-1./3.))+4.26*(T1D**(1./3.))-4.18
C
      F2=(0.5-1.87*T1D)+(5.*T1D-1.6)*(CHI**3)+(1.7-12.*T1D)*CHI
C
      F3=0.32*(T1D**(5./6.))+0.48*(T1D**(1./3.))*SIN3
C
      NF=(F1+0.63*F2)*(VC**(-4./3.))*(SD**(-5./12.))*(T2D**(-7./12.))
C
      NN=F3*(T2D**(-1.))*(VC**(-4./3.))
C
C  DETERMINE WHICH COMPONENT CONTROLS
C
      IF ( NF.GT.NN ) THEN
         PLP=NF
      ELSE
         PLP=NN
      ENDIF
C
C  RESET HOLDERS
C
      DIA1=DIA2
      DIA2=DIA
      PLP1=PLP2
      PLP2=PLP
C
C  CHECK IF PLP IS WITHIN TOLERANCE,IF NOT ITERATE
C
      TEST = PLP/SPF
C
C
      IF ( TEST.LT.0.99 .OR. TEST.GT.1.01 ) GO TO 100
C
C  IF SWITCH IS TRUE, ESTIMATE THE CRITICAL DIAMETER USING LINEAR
C  INTERPOLATION. THE TWO POINTS USED ARE THE ORIGIN AND THE VALUE
C  CALCULATED AT 11800 .
C
      IF ( SWITCH ) THEN
C
         DIA=DIA2*VHOLD/11800.0
         SWITCH=.FALSE.
         DIA1=0.0
         PLP1=0.0
C
```

```
      END IF
C
      RETURN
C
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE WILKIN (ANGR,DENS,DIA,EVWTHK,ITYPE,SHTHK,STAND,VELM,
     .                   WILKC )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  WILKIN DETERMINES THE DIAMETER THAT JUST PENETRATES THE VESSEL
C  WALL. IT IS BASED ON J.P.D. WILKINSON'S PAPER 'A PENETRATION
C  CRITERION FOR DOUBLED-WALLED STRUCTURE SUBJECT TO METEOROID IMPACT'
C  ,AIAA JOURNAL OCT 1969.
C
C  THE MAJOR ASSUMPTION USED BY WILKINSON IS THAT THE PROJECTILE
C  VAPORIZES ON IMPACT
C
C
C  VARIABLE LIST
C
C      ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL, RADIANS
C      CONST = MATERIAL CONSTANT DEFINED BY WILKINSON
C      C1 = INTERMEDIATE VARIABLE
C      DENS = ARRAY CONTAINING VALUES FOR DENSITY OF THE SHIELD &
C             VESSEL WALL MATERIALS
C      DIA = PROJECTILE DIAMETER (IN)
C      DIAM = PROJECTILE DIAMETER (CM )
C      EVWTHK = EQUIVALENT VESSEL WALL THK.  ( IN)
C      ITYPE = ANALYSIS TYPE  1=DEBRIS & METEOROIDS, 2=METEOROIDS
C      MASS = PROJECTILE MASS ( GRAMS )
C      MB = INTERMEDIATE VARIABLE
C      MT =       "          "
C      PI = PI
C      PROJDEN = PROJECTILE DENSITY ( G/CC )
C      RMI = UNIT MASS OF SHIELD (G/CM**2)
C      RM2 = UNIT MASS OF VESSEL WALL (G/CM**2)
C      SHDEN = SHIELD & VESSEL WALL DENISITY ( GRAMS/CC )
C      SHTHK = SHIELD THK. (IN)
C      STAND = SHIELD STAND-OFF DISTANCE  (IN)
C      STANDM = STAND IN CM.
C      VELM = VEL IN KM/SEC
C      VNORM = NORMAL COMPONENT OF THE VELEOCITY VECTOR,KM/SEC
C      VWDEN = VESSEL WALL DENSITY (LB/IN**3)
C      WILKC = ARRAY CONTAINING VALUES OF WILKINSON'S CONSTANT
C              FOR THE SHIELD & VESSEL WALL MATERIALS
C
C
C
      DIMENSION DENS(3),WILKC(3)
C
      INTEGER*2 ITYPE
C
      REAL MB,MT,MASS
C
C SET INITIAL VALUES
C
      PI = 3.141592654
C
C  SET PROJECTILE DENSITY IN G/CC
C
      PROJDEN = DENS(3)*27.705
C
C  SET SHIELD AND VESSEL WALL DENSITY IN G/CC
C
```

```
      SHDEN = DENS(1) * 27.705
      VWDEN = DENS(2) * 27.705
C
C  CONST IS A MATERIAL VARIABLE DEFINED IN THE PAPER & IS
C  EQUAL TO .401 FOR 2219 AL
C
      CONST = WILKC(2)
C
C  DETERMINE SHIELD & VESSEL WALL MASS PER UNIT AREA
C
      RM1 = SHTHK * 2.54 * SHDEN
      RM2 = EVWTHK * 2.54 * VWDEN
C
C  CONVERT STAND TO CM
C
      STANDM = STAND * 2.54
C
C  CALCULATE THE NORMAL COMPONENT OF THE VELOCITY VECTOR
C
      VNORM=VELM*COS(ANGR)
C
C  DETERMINE CRITICAL PROJECTILE DIAMETER
C
      MT = 1.44*(PI/6.0)**(1./3.)*CONST*RM1*RM2*STANDM**2.0
      MB = PROJDEN**(2./3.)*VNORM
      MASS = (MT/MB)**.75
      DIAM = (6.0*MASS/(PI*PROJDEN))**(1./3.)
C
C     CHECK IF APPROPRIATE EQ. WAS USED
C
      C1 = RM1 / ( PROJDEN * DIAM )
      IF ( C1 .GT. 1.0 ) THEN
C
C   WRONG EQN. USED,RECALC USING CORRECT EQN.
C
          MASS= 1.44*CONST*RM2*(STANDM**2.)/VNORM
          DIAM = (6.0*MASS/(PI*PROJDEN))**(1./3.)
      END IF
C
C  CONVERT DIAMETER TO IN
C
      DIA = DIAM / 2.54
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE REGRESS ( ANGR,DIA,MLI,SHTHK,STAND,VELM,VWTHK )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C  Subroutine REGRESS determines the critical projectile diameter for
C  a two-plate structure impacted by aluminum spherical projectiles. It
C  was developed from test data obtained during the NASA contract
C  'Integrated Space Station Wall Design Guide and Penetration Control
C  Plan'. The data varied from 2 to 8 km/sec.
C
C
C  Variable List
C
C   angr = impact angle measured from the normal, radians
C   dia = critical projectile diameter, in
C   d13 = intermediate variable
C   high = holder, last diameter to penetrate , in
C   ic = counter
C   low = holder, last diameter to not penetrate, in
C   lsd = intermediate variable
C   mli = logical variable indicating wether multi-layer insulation is
C         used
C   mt = intermediate variable
C   np = number of plates penetrated, excluding the shield
C   shthk = shield thickness , in
C   stand = shield stand-off , in
C   switch = logical variable, used to determine if a penetrating
C            diameter has been determined
C   tt  = intermediate variable
C   t13 =      "         "
C   vc2 =      "         "
C   velm = impact velocity, km/sec
C   vwthk = vessel wall thickness , in
C
C
C
      INTEGER IC
C
      REAL LOW,LSD,MT,NP
C
      LOGICAL MLI,SWITCH
C
C  Intialize Variables
C
      IC=0
      LOW=0.0
      HIGH=5.0
      SWITCH=.TRUE.
C
C  Set MLI constant
C
      IF ( MLI ) THEN
         MT=-14.0
      ELSE
         MT=0.0
      END IF
C
C  Since the equation does not solve for the critical projectile
```

```
C  diameter directly, use a binary search technique to determine it.
C  First determine a diameter that penetrates then narrow in on the
C  actual diameter.
C
  100 CONTINUE
C
      IC=IC+1
C
      IF ( SWITCH ) THEN
         HIGH=HIGH*2.0
      END IF
C
      DIA=(HIGH+LOW)/2.0
C
C  Check that the diameter is less then 50 cm , if not stop
C
      IF ( DIA.GT.50.0 ) THEN
         WRITE ( 6,'(''---ERROR--- Diameter greater than 50 cm in '',
     1           ''subroutine REGRESS'')')
         STOP
      END IF
C
C  Calculate the intermediate variables
C
      D13=DIA**(1./3.)
      LSD=LOG10(STAND)/DIA
      T13=SHTHK**(1./3.)
      TT=TAN(ANGR)
      VC2=VELM*(COS(ANGR)**2)
C
C  Calculate the number of plates penetrated
C
      NP=1.52-6.18*T13-18.8*VWTHK-0.146*LSD+MT*SHTHK+10.8*D13-0.287*VC2
     1   -0.713*TT
C
C  Check for convergance
C
      IF ( IC.GT.100 ) THEN
         WRITE ( 6,'(''---ERROR--- REGRESS failed to converge after '',
     1           ''100 cycles'' )')
         STOP
      END IF
C
C  Has a diameter that penetrates been found, if not reset holders and
C  try again
C
      IF ( SWITCH ) THEN
         IF ( NP.GT.1 ) THEN
            SWITCH=.FALSE.
            HIGH=DIA
            GO TO 100
         END IF
      END IF
C
C  Does the diameter yeild an acceptable result, if not rest holders
C  and try again
C
      IF ( NP.LT.0.999 ) THEN
         LOW=DIA
         GO TO 100
      ELSE IF ( NP.GT.1.001 ) THEN
         HIGH=DIA
         GO TO 100
      END IF
C
C  Finished
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
         SUBROUTINE BALLIST (ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                           INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  BALLISTIC DETERMINES THE DIAMETER THAT JUST PENETRATES. IT ULTILZES
C  THE BALLISTIC PORTION OF BOEING'S HYPERVELOCITY CODE PEN4.
C
C
C  VARIABLE LIST
C
C    ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL,RADIANS
C    BHARD = ARRAY CONTAINING THE VALUES OF THE BRINNEL HARDNESS FOR
C            THE SHIELD & VESSEL WALL MATERIALS
C    C = ARRAY CONTAINING VALUES FOR THE SPEED OF SOUND FOR THE
C        SHIELD AND VESSEL WALL MATERIALS (FT/SEC)
C    DEN = PROJECTILE DENSITY (LB/CFT)
C    DENS = ARRAY CONTAINING VALUES FOR THE DENSITY FOR THE
C           SHIELD AND VESSEL WALL MATERIALS (LB/IN**3)
C    DIA = SPHERICAL PROJECTILE DIAMETER (IN)
C    DIA1 = PREVIOUS DIAMETER THAT PENETRATED (IN)
C    DIA2 = DIAMETER THAT PENETRATED TWO ITERATIONS PREVIOUSLY (IN)
C    EVWTHK = EQUIVALENT VESSEL WALL THICKNESS,IN.
C    FY = ARRAY CONTAINING THE VALUES OF THE YIELD STRENGTH FOR THE
C         SHIELD & VESSEL WALL MATERIALS
C    FSU = ARRAY CONTAINING VALUES OF THE ULTIMATE SHEAR STRENGTH
C          FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C    FTU = ARRAY CONTAINING VALUES OF THE ULTIMATE TENSILE STRENGTH
C          FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C    I = COUNTER
C    INITIAL = LOGICAL PARAMETER USED TO DETRMINE IF CURRENT CALL
C             IS INITIAL ONE.
C    MASS = PROJECTILE MASS,LBS
C    N = INCREMENT MULIPLIER
C    PDENS = DENS ARRAY CONVERTED TO SLUGS/FT**3
C    PFY = FY ARRAY CONVERTED TO PSF
C    PFSU = FSU ARRAY CONVERTED TO PSF
C    PFTU = FTU ARRAY CONVERTED TO PSF
C    P1 = LAST MASS GUESS TO NOT PENETRATE
C    P2 = LAST MASS GUESS TO PENETRATE
C    PEN = TRUE OR FALSE
C    PI = 3.14
C    PMINCR = INITIAL MASS GUESS INCREMENT
C    RATIO = P2 / P1
C    SHATTER = LOGICAL PARAMETER USED TO DETERMINE IF PEN4 BALLISTIC
C              LIMITS HAVE BEEN EXCEEDED
C    SHPV = ARRAY CONTAING VALUES OF THE SHOCK PROJECTILE VELOCITY
C           FOR THE SHIELD AND VESSEL WALL MATERIALS,( UNITLESS )
C    SHTHK = SHIELD THICKNESS,IN.
C    SPACE = ARRAY CONTAINING THE SHIELD SPACING ,FT.
C    STAND = SHIELD STAND-OFF,IN.
C    TARMAT = ARRAY CONTAINING MATERIAL PROPERTY POINTERS
C    THETA = IMPACT ANGLE (RAD),MEASURED FROM THE NORMAL
C    THICK = SHIELD & VESSEL WALL THICKNESS,FT.
C    VELE = COLLISION VELOCITY,FT/SEC
C    VEL1 = VEL FOR DIA1
C    VEL2 = VEL FOR DIA2
C    V0 = IMPACT VELOCITY (FT/SEC)
C    VR2 = RESIDUAL VELOCITY AFTER VESSEL WALL,FT/SEC
C
```

```
C
C
      DIMENSION BHARD(3),C(3),DENS(3),FY(3),FSU(3),FTU(3),PDENS(3),
     .          PFY(3),PFSU(3),PFTU(3),SHPV(3),SPACE(2),THICK(3),
     .          TARMAT(3)
C
      INTEGER PROJMAT,TARMAT
C
      LOGICAL PEN,INITIAL,SHATTER
C
      REAL MASS
C
      SAVE DIA1,DIA2,P2,VEL1,VEL2
C
C
C  SET INITIAL VALJES
C
      I = 0
      N = 0
      PEN = .FALSE.
      PMINCR = 1.0E-04
      PI = 3.1415926536
      P1 = 1.0E-06
C
C  CONVERT VARIABLES TO PEN4 FORMAT AND UNITS
C
      DO 50 I=1,3
C
C  CONVERT DENS TO SLUGS/FT**3
C
         PDENS(I)=DENS(I)*1728./32.2
C
C  CONVERT FSU AND FTU TO PSF
C
         PFY(I)=FY(I)*144.
         PFTU(I)=FTU(I)*144.
         PFSU(I)=FSU(I)*144.
C
50    CONTINUE
C
      PROJMAT = 3
      SPACE(1) = STAND/12.0
      SPACE(2) = 0.0
      TARMAT(1) = 1
      TARMAT(2) = 2
      TARMAT(3) = 0
      THETA = ANGR
      THICK(1) = SHTHK/12.0
      THICK(2) = VWTHK/12.0
      THICK(3) = 0.0
      V0 = VELE
C
C  PEN4 DOES NOT SOLVE FOR THE CRITICAL PROJECTILE DIAMETER DIRECTLY
C  THEREFORE AN ITERATIVE APPROACH IS TAKEN. PEN4 ONLY RETURNS
C  A LOGICAL PARAMETER INDICATING IF PENETRATION HAS OCCURED.
C
C
C  FOR INITIAL CASE SET P2 > 0.0
C
      IF(INITIAL)THEN
        P2 = 1.0E-04
      ENDIF
C
```

```
C  IF SHATTER HAS OCCURED NO NEED TO CALL PEN4,USE LINEAR APPROX.
C
      IF (SHATTER) GO TO 500
C
C  DETERMINE INITIAL MASS THAT PENETRATES
C
100   MASS = P2 + N * PMINCR
C
      NC = NC + 1
C
      CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,PROJMAT,SHPV,
     .           C,SPACE,TARMAT,THICK,PFY,*500)
C
      IF ( PEN ) THEN
C
C    IF PENETRATION TRY WITH MASS BETWEEN P1 & P2
C
         PEN = .FALSE.
         P2 = MASS
         MASS = ( P1 + P2 ) / 2.0
C
         NC = NC + 1
         CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,PROJMAT,
     .              SHPV,C,SPACE,TARMAT,THICK,PFY,*500)
C
C
200      IF ( PEN ) THEN
C
C      IF PENETRATION,SET P2 = MASS & CHECK RATIO
C
            PEN = .FALSE.
            P2 = MASS
            RATIO = P2 /P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,
     .                     PROJMAT,SHPV,C,SPACE,TARMAT,THICK,PFY,
     .                     *500)
               GO TO 200
            ELSE
C
C        IF FALSE,MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
C
         ELSE
C
C      IF FALSE,SET P1 = MASS AND CHECK RATIO
C
            P1 = MASS
            RATIO = P2 / P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE,TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,
     .                    PROJMAT,SHPV,C,SPACE,TARMAT,THICK,PFY,
     .                    *500)
               GO TO 200
            ELSE
C
C        IF FALSE,MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
         ENDIF
      ELSE
C
C    IF FALSE INCREASE INCREMENT & TRY AGAIN
C
         P1 = MASS
         I = I + 1
         N = 2 ** I
         GO TO 100
      END IF
C
      P2=MASS
C
C  CALCULATE DIAMETER
C
C
C
300   DEN = DENS(3)*1728.0
C
      DIA = ((MASS*6.0)/(PI*DEN))**(1.0/3.0)
C
C  CONVERT TO IN.
C
      DIA = DIA * 12.0
C
C
C
C  RESET HOLDERS
C
      DIA2=DIA1
      DIA1=DIA
      VEL2=VEL1
      VEL1=VELE
C
C
      RETURN
C
C  SINCE SHATTER HAS OCCURED NO NEED TO CALL PEN4 USE LINEAR APPROXL
C
500   SLOPE=(DIA1-DIA2)/(VEL1-VEL2)
      DIA=DIA2+SLOPE*(VELE-VEL2)
      RETURN
C
      END
```

```
C
      SUBROUTINE OPEN4 (Penetration,Mass,Theta,SHATTER,V0,
     1                 Density,Hardness,ProjMat,ShockProjVel,SoundVel,
     2                 Spacing,TarMat,Thick,YieldStrength,*)
C     This version of pen4 is strictly for use in the sub-shatter velocity
C       regime.  It is a modified version of spin14, created on 6/21/85.

      LOGICAL Penetration,SHATTER
      REAL Mass,LAMBDA,NR,NF,NH,J,NR2,NHT,LastPK,MR
      INTEGER ProjMat,TarMat,Exponent,TopCount,BottomCount1,BottomCount2
      DIMENSION Thick(3),TarMat(3),Spacing(2),
     :           Density(3),YieldStrength(3),SoundVel(3),
     :                 ShockProjVel(3),Hardness(3),Epsil(2),
     :           NR(3),NR2(3),RF(3),Flagit(3)
C
C           Density in Slugs/CubicFoot
C           YieldStrength in Lbs/SquareFoot
C           SoundVel in Feet/Second
C           Hardness is Brinell Scale
      DATA Epsil,Gamma/5.71,5.71,90.0/
      DATA F1,F2/4.0,1.0/
      DATA EffectiveThicknessRatio/.6/
      DATA A,B/2.0,3.125E-04/
      PI=3.141592653589793
      Small = 1E-36
      RecipSqrt2PI=1./SQRT(2.*PI)
C     **************Calculate Radius of Projectile Sphere*************
      RP=(Mass*3./
     1              (Density(ProjMat)*32.2*4.*PI))**(1./3.)
      Diam = RP * 2.
C     *************************************************************

C     ************Compute ResidualVel********************************
      VDEL=( V0**2*1.33*RP**2*Density(ProjMat)-
     1 F1*YieldStrength(TarMat(1))*THICK(1)**2/COS(THETA)**2
     2 *A*EXP(-B*V0) )
     2 /( 1.33*RP**2*Density(ProjMat)+F2*RP*THICK(1)*
     3 Density(TarMat(1))/COS(THETA) )
      VR=SQRT(AMAX1(VDEL,0.))
C     *************************************************************

C     ************Cratering V50**************************************
      V50=SQRT((Thick(1)*EffectiveThicknessRatio/COS(Theta)/
     1  (0.281*Diam*(Density(ProjMat)/Density(TarMat(1)))**(1./3.)))
     2  **(1./.31)/Density(ProjMat)*(2.*YieldStrength(TarMat(1))))
C     *************************************************************

C     **************IF No Penetration Report Result and Exit*********
      IF (V0.LT.V50) THEN
            Penetration = .FALSE.
            RETURN
      ELSE IF (Thick(2).EQ.0.0) THEN
            Penetration = .TRUE.
            RETURN
      ENDIF
C     *************************************************************

C     **************Mass Loss Regime Decision***********************
      ToverD = Thick(1)/Diam
      FTHETA=-2.433E-4 - 1.643E-2*COS(THETA)+3.201*COS(THETA)**2-
     :    2.184*COS(THETA)**3
      IF (ToverD.LT..40) THEN
            Vf = 4100.
      ELSE
            Vf = 4986.*ToverD**.21
      END IF
      IF (V0.LT.VF+4000.) THEN
        IF (V0.LT.Vf) THEN
              CALL MassErr(V0,Mass,Hardness(ProjMat),
     :         Density(ProjMat),Density(TarMat(1)),Thick(1),Diam,MR)
            ELSE
              CALL Fract(MR,V0,Vf,FTheta,Mass,ToverD)
            END IF
            CALL LarMR(VR,MR,V50,Thick(2),EffectiveThickNessRatio,Theta,
     :                 Diam,Density(ProjMat),Density(TarMat(2)),
     :                 YieldStrength(TarMat(1)),Penetration)
      ELSE
C       **********The rest of this subroutine contains the evaluation******
C       ************ of shatter regime multiple cratering penetration*******
C       ######Shatter regime removed for independent use of ballistic######
C       #################And fracture regime evaluation####################
        SHATTER = .TRUE.
            RETURN 1
      END IF
      RETURN
      END

      SUBROUTINE Fract(MR,V0,Vf,FTheta,MProj,ToverD)
      REAL MR,MProj
      IF (V0.GT.Vf+2000.) THEN
            MR = MProj*FTheta*.16
      ELSE IF (ToverD.GT..25) THEN
            MR = MProj*FTheta*.25
      ELSE
            MR = MProj*FTheta*.667
      END IF
      RETURN
      END
      SUBROUTINE LarMR(VR,MR,V50,Thickness,
     :                 EffectiveThicknessRatio,Theta,Diam,ProjDensity,
     :                 TargetDensity,TargetYieldStrength,Penetration)
      REAL MR
      LOGICAL Penetration
      V50=SQRT((Thickness*EffectiveThicknessRatio/COS(Theta)/
     1  (0.281*Diam*(ProjDensity/TargetDensity)**.33333))
     2  **(1./.31)/ProjDensity*(2.*TargetYieldStrength))
      IF (VR.LT.V50) THEN
            Penetration = .FALSE.
      ELSE
            Penetration = .TRUE.
      END IF
      RETURN
      END
      SUBROUTINE MassErr(V1,MFC1,BHNC,RhoP,RhoT,Thick,Diam,MR)
      REAL MR,MS1,MFC1
      RhoC = RhoP*.01873
      RhoS = RhoT*.01873
      T = Thick*12.
      DFC1 = Diam*12
C     DFC1 = SQRT(4*AC/PI)
      UFC=SQRT(2680*BHNC/RHOC)
      VP=V1/UFC
      AC1=PI*DFC1**2/(4*COS(THETA))
      MS1=RHOS*T*AC1
      MR=MFC1+0.5*MS1*LOG(2.74/(1+VP**2))
      MR = AMIN1(MR,MFC1)
      RETURN
      END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE SINGLE ( ANGR,DIA,VELE,VWTHK,DENS,FTU )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Single determines the critical projectile diameter for single plate
C  configurations. It is based on the Schmidt-Holsapple crater volume
C  equation. The equation was solved for the critical diameter as a
C  function of the plate and projectile properties in english units.
C  It assumes that a hemispherical crater depth equal to 70 percent of
C  the plate thickness causes failure. This is a attempt to account for
C  spall and was reccomended by Mike Bjorkman of the Boeing SHock Physics
C  group.
C
C
C  Ref: 'On Scaling of the Crater Dimensions 2. Impact Processes',
C       K.A. Holsapple & R.M. Schmidt, JGeophy Res, v87,nb3,10 March 1982,
C       p1849-70
C
C
C  Variable List
C
C    angr = impact angle measured from the normal, radians
C    dia = critical projectile diameter, in
C    dr = intermediate variable
C    fr =     "           "

C    vele = relative velocity of the projectile, ft/sec
C    vwthk = vessel wall thickness, in
C
C  Array list
C
C    dens = density, lb/in**3
C              1- shield
C              2- vessel wall
C              3- projectile
C    ftu = ultimate tensile strength, psi
C              1- shield
C              2- vessel wall
C              3- projectile
C
C
      DIMENSION DENS(3),FTU(3)
C
      DR=(DENS(3)/DENS(2))**(-0.159)
      V2=(VELE*COS(ANGR))**2
      FR=(2.6833*FTU(2)/DENS(3)/V2)**0.236
C
      DIA=1.442*DR*FR*VWTHK
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE PEN4 ( ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                  INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  PEN4 DETERMINES THE DIAMETER THAT JUST PENETRATES. IT ULTILZES
C  THE UPDATED  BOEING'S HYPERVELOCITY CODE PEN4 VERSION 10.
C
C
C  VARIABLE LIST
C
C     ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL , RADIANS
C     BHARD = ARRAY CONTAINING THE VALUES OF THE BRINNEL HARDNESS FOR
C             THE SHIELD & VESSEL WALL MATERIALS
C     C = ARRAY CONTAINING VALUES FOR THE SPEED OF SOUND FOR THE
C         SHIELD AND VESSEL WALL MATERIALS (FT/SEC)
C     DEN = PROJECTILE DENSITY (LB/CFT)
C     DENS = ARRAY CONTAINING VALUES FOR THE DENSITY FOR THE
C            SHIELD AND VESSEL WALL MATERIALS (LB/IN**3)
C     DIA = SPHERICAL PROJECTILE DIAMETER (IN)
C     DIA1 = PREVIOUS DIAMETER THAT PENETRATED (IN)
C     DIA2 = DIAMETER THAT PENETRATED TWO ITERATIONS PREVIOUSLY (IN)
C     EVWTHK = EQUIVALENT VESSEL WALL THICKNESS , IN.
C     FY = ARRAY CONTAINING THE VALUES OF THE YIELD STRENGTH FOR THE
C          SHIELD & VESSEL WALL MATERIALS
C     FSU = ARRAY CONTAINING VALUES OF THE ULTIMATE SHEAR STRENGTH
C           FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C     FTU = ARRAY CONTAINING VALUES OF THE ULTIMATE TENSILE STRENGTH
C           FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C     I = COUNTER
C     INITIAL = LOGICAL PARAMETER USED TO DETRMINE IF CURRENT CALL
C               IS INITIAL ONE.
C     MASS = PROJECTILE MASS , LBS
C     N = INCREMENT MULIPLIER
C     P1 = LAST MASS GUESS TO NOT PENETRATE
C     P2 = LAST MASS GUESS TO PENETRATE
C     PEN = TRUE OR FALSE
C     PI = 3.14
C     PMINCR = INITIAL MASS GUESS INCREMENT
C     RATIO = P2 / P1
C     SHATTER = LOGICAL PARAMETER USED TO DETERMINE IF PEN4 BALLISTIC
C               LIMITS HAVE BEEN EXCEEDED
C     SHPV = ARRAY CONTAING VALUES OF THE SHOCK PROJECTILE VELOCITY
C            FOR THE SHIELD AND VESSEL WALL MATERIALS , ( UNITLESS )
C     SHTHK = SHIELD THICKNESS , IN.
C     SPACE = ARRAY CONTAINING THE SHIELD SPACING ,FT.
C     STAND = SHIELD STAND-OFF , IN.
C     TARMAT = ARRAY CONTAINING MATERIAL PROPERTY POINTERS
C     THETA = IMPACT ANGLE (DEG) , MEASURED FROM THE NORMAL
C     THICK = SHIELD & VESSEL WALL THICKNESS , FT.
C     VELE = COLLISION VELOCITY , FT/SEC
C     VEL1 = VEL FOR DIA1
C     VEL2 = VEL FOR DIA2
C     V0 = IMPACT VELOCITY (FT/SEC)
C     VR2 = RESIDUAL VELOCITY AFTER VESSEL WALL , FT/SEC
C
C
C
      DIMENSION BHARD(3),C(3),DENS(3),FY(3),FSU(3),FTU(3),PDENS(3),
     .          PFY(3),PFSU(3),PFTU(3),SHPV(3),SPACE(2),THICK(3),
     .          TARMAT(3)
C
      INTEGER PROJMAT,TARMAT
C
      LOGICAL PEN,INITIAL,SHATTER
C
      REAL MASS
C
      SAVE DIA1,DIA2,P2,VEL1,VEL2
C
C
C  SET INITIAL VALUES
C
      I = 0
      N = 0
      PEN = .FALSE.
      PMINCR = 1.0E-04
      PI = 3.1415926536
      P1 = 1.0E-06
C
C  CONVERT VARIABLES TO PEN4 FORMAT AND UNITS
C
C
      PROJMAT=2
      SPACE(1) = STAND/12.0
      SPACE(2) = 0.0
      TARMAT(1) = 1
      TARMAT(2) = 2
      TARMAT(3) = 0
      THETA = ANGR*180.0/PI
      THICK(1) = SHTHK/12.0
      THICK(2) = VWTHK/12.0
      THICK(3) = 0.0
C
C  PEN4 DOES NOT SOLVE FOR THE CRITICAL PROJECTILE DIAMETER DIRECTLY
C  THEREFORE AN ITERATIVE APPROACH IS TAKEN. PEN4 ONLY RETURNS
C  A LOGICAL PARAMETER INDICATING IF PENETRATION HAS OCCURED.
C
C
C  FOR INITIAL CASE SET P2 > 0.0
C
      IF(INITIAL)THEN
         P2 = 1.0E-04
      ENDIF
C
C  IF SHATTER HAS OCCURED NO NEED TO CALL PEN4 , USE LINEAR APPROX.
C
C
C  DETERMINE INITIAL MASS THAT PENETRATES
C
100   MASS = P2 + N * PMINCR
C
      NC = NC + 1
C
      CALL NPEN4 ( VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,PROJMAT,
     1            DENS,FTU,C )
C
      IF ( PEN ) THEN
C
C    IF PENETRATION TRY WITH MASS BETWEEN P1 & P2
C
         PEN = .FALSE.
         P2 = MASS
         MASS = ( P1 + P2 ) / 2.0
C
         NC = NC + 1
         CALL NPEN4 ( VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,PROJMAT,
     1               DENS,FTU,C )
C
C
200      IF ( PEN ) THEN
C
C      IF PENETRATION , SET P2 = MASS & CHECK RATIO
C
            PEN = .FALSE.
            P2 = MASS
            RATIO = P2 /P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL NPEN4 (VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,
     1                    PROJMAT,DENS,FTU,C )
               GO TO 200
            ELSE
C
C        IF FALSE , MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
C
         ELSE
C
C      IF FALSE , SET P1 = MASS AND CHECK RATIO
C
            P1 = MASS
            RATIO = P2 / P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE , TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL NPEN4 (VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,
     1                    PROJMAT,DENS,FTU,C)
               GO TO 200
            ELSE
C
C        IF FALSE , MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
         ENDIF
      ELSE
C
C    IF FALSE INCREASE INCREMENT & TRY AGAIN
C
         P1 = MASS
         I = I + 1
         N = 2 ** I
         GO TO 100
      END IF
C
      P2=MASS
```

```
C
C  CALCULATE DIAMETER
C
C
C
300   DEN = DENS(3)*1728.0
C
      DIA = ((MASS*6.0)/(PI*DEN))**(1.0/3.0)
C
C  CONVERT TO IN.
C
      DIA = DIA * 12.0
C
C
C
C  RESET HOLDERS
C
      DIA2=DIA1
      DIA1=DIA
      VEL2=VEL1
      VEL1=VELE
C
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE NPEN4 (Vil,MProj,Thetal,Thickl,Space,Pennon,
     :       Shater,PrMatl,Densel,YStrnl,SoundV)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C     PrMat Integer Projectile Material 1:Steel, 2:Aluminum ,3:Ice

      CHARACTER Shape*3
      LOGICAL Pennon,Shater
      INTEGER Maxk(5)
      INTEGER PrMat,PrMatl,TarMat(10),TMatSp(10),Plate,Bin,NBin,I
      REAL RF(5),RC(5)
      REAL NF,J,MR,MProj,LastSp,LRM,FrMass(5)
      REAL Diam,Vi,Vr,ViLRM,VrLRM,Epsil,Gamma,Vil,Pi,Theta,SumSp
      REAL A,B,D,R,X,Y,ToverD,Rh,PlateM,FTheta,AllMas,Vc,DelJ,DelJ2
      REAL P,EffP,Pet,Pgrady,Thetal,AvgMas,Rp,F1,Vf
      REAL      Thick(10),Space(9),Thickl(10),
     :          PDense(3),PYStrn(3),PSondV(3),FrTuff(3),
     :          Dense(10),YStren(10),SoundV(10)
      REAL Densel(10),YStrnl(10)
      REAL ViX,MrMax,MProjX
      DOUBLE PRECISION Intact,HoArea
      DOUBLE PRECISION SumPr(5)
      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk,As,Ac

      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)
      COMMON /Crater/As,Ac(5),P(5)
C                                    Steel   Aluminum   Ice
      DATA PDense/                   15.11 ,     5.39 ,   1.94/
      DATA PYStrn/                 8.35E+07 , 5485000.,   0.0/
      DATA PSondV/                   14960. ,   17569.,   0.0/
      DATA FrTuff/                      36. ,      39.,   39./
C       PrMat  Projectile Material 1:Steel, 2:Aluminum ,3:Ice
C       Calculation units
C          Dense  in Slugs/CubicFoot       Density
C          YStren in Lbs/SquareFoot        Uniaxial Ultimate Stress
C          SoundV in Feet/Second           Speed of Sound
C          Theta  in Radians               Impact Angle
C          FrTuff in MegaPascals Meter^.5  Fracture Toughness
C       Input units
C          Densel  input in Lbs/CubicInch
C          YStrnl  input in PSI
C          SoundV  input in Feet/Sec
C          Vil     Feet/Sec             Impact Velocity
C          MProj   Pounds               Projectile Mass
C          Thetal  Degrees              Striking Angle
C          Thick   Feet                 Target Plate Thickness
C          Space   Feet                 Spacing Between Target Plates
C        Outputs
C          Pennon Logical              Penetration Flag
C          Shater Logical              Fragment Shatter Flag

      DATA Gamma/1.5708/
      DATA F1/4.0/
      DATA NBin/5/
      DATA TarMat/1,2,3,4,5,6,7,8,9,10/
      DATA TMatSp/2,2,2,2,2,2,2,2,2,2/
      DO 10 Plate=1,10
```

```
        Thick(Plate)=Thick1(Plate)
        Dense(Plate)=Dense1(Plate)/32.2*1728.
        YStren(Plate)=YStrn1(Plate)*144.
 10   CONTINUE
      Theta = Theta1/57.3
      PrMat = PrMat1
C     END DO
      Vi=Vi1
      PI=3.141592653589793
      Shater =.FALSE.
C     **************Calculate Radius of Projectile Sphere*************
      RP=(MProj*3./(PDense(PrMat)*32.2*4.*PI))**(1./3.)
      Diam = RP * 2.
C     ***************************************************************

      DO 2040 Plate=1,10
        IF (.NOT.Shater) THEN
C         *** This option is for single penetrators************************
C         **Compute Residual Velocity

          Call ResVel
     :           (Vr,Vi,RP,PDense(PrMat),Dense(TarMat(Plate)),
     :       Thick(Plate),Theta,PrMat,TMatSp(Plate))

C         **************IF No Penetration Report Result and Exit**********
          IF (Vr.EQ.0.) THEN
            Pennon = .FALSE.
            RETURN
          ELSE IF (Thick(Plate+1).EQ.0.0.OR.Plate.EQ.10) THEN
            Pennon = .TRUE.
            RETURN
          ENDIF
C         ***************************************************************
C         **************Mass Loss Regime Decision BAC IR&D****************
          ToverD = Thick(Plate)/Diam
          IF (PrMat.EQ.2.AND.TMatSp(Plate).EQ.2) THEN
            IF (ToverD.LT..1) THEN
              Vf = 1116*ToverD**(-.55)
            ELSE
              Vf = 4757*ToverD**.08
            END IF
          ELSE IF (PrMat.EQ.1.AND.TMatSp(Plate).EQ.2.) THEN
            IF (Shape.EQ.'CYL') THEN
              Vf=5020* ToverD**.4
            ELSE IF (Shape.EQ.'CUB') THEN
              Vf = AMAX1(1450*ToverD**(-.39),
     :                   4561*ToverD**42*(Diam*12*2.54)**(-.33))
            ELSE
              Vf = AMAX1(1450*ToverD**(-.39),
     :                   4561*ToverD**.42*(Diam*12*2.54)**(-.33) )

C              Vf = AMAX1(2362*ToverD**(-.35)*(Diam*12*2.54)**(-.32),
C      :                  3937*ToverD**.23   *(Diam*12*2.54)**(-.25))
            END IF
          ELSE IF (PrMat.EQ.1.AND.TMatSp(Plate).EQ.1.) THEN
            Vf = 7021*ToverD**.39
          END IF
C         ***************************************************************
C          RH=RP*(1.372E-4*Vi*(THICK(Plate)/(2.*RP))**(2./3.)+.9)*
C      :      (1-EXP(-(1.48-Theta)/.166))
          RH = .5*Thick(Plate)*11.02*(1-EXP(-(1.48-Theta)/.166))*
     :      (1-EXP(-(PDense(PrMat)*Vi**2/YStren(TarMat(Plate)))**.415
     :      *(PDense(PrMat)/Dense(TarMat(Plate)))**(-.15)/ToverD/29.9))
```

```
          RH = AMAX1(RH,RP)
          IF (Vi.GT.Vf) THEN
            SumSp=Space(Plate)
            Shater=.TRUE.
C           *** Compute plate spall ****************************************
            PlateM=PI*RH**2/COS(Theta)*Thick(Plate)*
     :                  Dense(TarMat(Plate))*32.2
C           ****************************************************************
C           *********Hydrocode Predicted Mass Loss Due to Impact Angle******
            FTHETA=-2.433E-4 - 1.643E-2*COS(THETA)+3.201*COS(THETA)**2-
     :       2.184*COS(THETA)**3
C           ****************************************************************
C           *** COMPUTE FRAGMENT NUMBERS************************************
            IF (PrMat.EQ.3) THEN
              MProj=PlateM
              RP=(PlateM*3./(Dense(TarMat(Plate))*32.2*4.*PI))**(1./3.)
            END IF

            CALL MasChr (Vi,MProj,ToverD,Theta,RP,PDense(PrMat),
     :                   Dense(TarMat(Plate)),PSondV(PrMat),NBin,
     :                   RF,NR,LRM,AvgMas,PrMat,FrTuff(PrMat))

            IF (PrMat.EQ.3) THEN
              PrMat=TMatSp(Plate)
            ELSE
              AllMas=PlateM+MProj
              DO 2010 I=1,NBin-1
                FrMass(I)=4./3.*PI*RF(I)**3*PDense(PrMat)*32.2
                NR(I)=NR(I)*AllMas/MProj
 2010         CONTINUE
            END IF
            Nr(NBin) = 1.D0
            ViLRM=Vr
C           ***********Calculate Spray Angle********************************
C            Vc =11155*ToverD**(-.52)
C            Epsil = 45*(1-EXP((-(Vi-Vf)/Vc)))/57.3
            Vc = 4889 * ToverD**(-.23)
            Epsil = 52*(1-EXP((-(Vi-Vf)/Vc)))/57.3
C           *** Assurance of ThetaR+Epsil<90 and Spray Area finite**********.
            ThetaR = AMIN1(Theta,1.41-Epsil)
C           *** Calculate Spray Area****************************************
            DELJ=RP/2.0*(COS(EPSIL)-TAN(ThetaR)*SIN(EPSIL))
            DELJ2=RP*(1.0-TAN(ThetaR)*TAN(EPSIL))
            J=Space(Plate)*SIN(GAMMA)/SIN(ThetaR+GAMMA)
            X=(J*SIN(EPSIL)/2.+DELJ)*(1./SIN(ThetaR+GAMMA+EPSIL)-
     :        1./SIN(ThetaR+GAMMA-EPSIL))
            Y=X*SIN(ThetaR+GAMMA)
            R=(J-X*COS(ThetaR+GAMMA))*TAN(EPSIL)+DELJ2
            B=SQRT(R**2-Y**2)
            A=(J*SIN(EPSIL)/2.+DELJ)*(1./SIN(ThetaR+GAMMA-
     :        EPSIL)+1./SIN(ThetaR+GAMMA+EPSIL))
            AS=PI*A*B
C           *********************************************************
C           *** Allowance for increased penetration due to spalling of ****
C           *** next plate ************************************************
              EffP=1.7
C           ****************************************************************
          ELSE
            IF (PrMat.EQ.3) THEN
              MProj=3.1415926*Rh**2*Thick(Plate)
     :                     *Dense(TarMat(Plate))*32.2
              PrMat=TMatSp(Plate)
            END IF
            Vi=Vr
          END IF
C         ****************************************************************
        ELSE

C         **********The rest of this subroutine contains the evaluation*
C         *********** of shatter and fracture regime multiple cratering*

C         *** Allowance for increased penetration due to spalling of ****
C         *** next plate *************************************************
          Thick(Plate) = Thick(Plate)/EffP
C         *****************************************************************

C         *** COMPUTE PLATE CRATER Depth and Radius************************
          DO 2030 I=1,NBin
            D=2.*RF(I)
            PGrady=0.281*D*(PDense(PrMat)/Dense(TarMat(Plate)))**(1./3.)
     :         *(PDense(PrMat)*(Vr*COS(Theta))**2/
     :          (2.*YStren(TarMat(Plate))))**0.31
            IF (PrMat.EQ.1) THEN
              Pet = D*(Vr*COS(Theta)/12468.)**1.32
              P(I) = AMIN1(Pet,PGrady)
              IF (Vr.LT.3.9*3281) THEN
                RC(I) = P(I)/2.*(Vr/3281./3.8)**(-1.32)
              ELSE
                RC(I) = P(I)/2.*(Vr/3281./4.6)**.2
              END IF
            ELSE
              Pet = D*2.33E-5*(Vr*COS(Theta))**1.16
              P(I)= AMIN1(Pet,PGrady)
              RC(I)=P(I)/(1-EXP(-Vr/5578.))
            END IF
C           ********************************************************

C           *** COMPUTE AVERAGE IMPACTS WITHIN CRATER**************
            AC(I)=PI*RC(I)**2/F1
            AC(I)=DMIN1(Ac(I),.999999999999D0*AS)
            PCR(I)=DMIN1(1D0,AC(I)/AS)
            LAMBDA(I)=NR(I)*AC(I)/AS
            SigSq(I) = Lambda(I)*(1D0-PCR(I))
            Sigma(I) = SQRT(SigSq(I))
C           ********************************************************
 2030     CONTINUE
C         *** New Version eight section**************************
C         *** This subroutine finds the minimum number of each***
C         *** size particle that must impact in one crater to ***
C         *** penetrate the plate ,how many craters they are in,*
C         *** and how many fragments are involved in shallower***
C         *** craters.*******************************************
          CALL PenK(Plate,Thick,NBin,Maxk)
          CALL Countr
     :          (NBin,P,Thick(Plate),Maxk,Intact,Nr,Ac,As)
C         *** Number and Area of Holes and Residual Particles****
          HoArea=As*(1.D0-Intact)
          PlateM = HoArea*Thick(Plate)*Dense(Plate)*32.2
          AllMas=0.
          DO 2110 I=1,NBin-1
            AllMas=FrMass(I)*Nr(I)+AllMas
 2110     CONTINUE
          IF (AllMas.GT.0) THEN
            DO 2210 I=1,NBin-1
              NR(I)=NR(I)*(1.+PlateM/AllMas)
 2210       CONTINUE
```

```
          END IF
C         *** Separate calculation for LRM****
C         *** This option is for single penetrators*************************

          Call ResVel
     :      (Vr,ViLRM,RF(NBin),PDense(PrMat),Dense(TarMat(Plate)),
     :      Thick(Plate),Theta,PrMat,TMatSp(Plate))

C     ** Convert V to Km/S and MProj to grams
          ViX = Vi/3281.
          MProjX = LRM * 454.

C     ** Largest Residual Mass **
          ToverD=Thick(Plate)/2./Rf(NBin)
          CALL RFMax(MProjX,PDense(PrMat),ViX,RF(NBin),ToverD,MRMax,
     :                PSondV(PrMat),PrMat,Vc,Theta,FrTuff(PrMat))
          LRM=MrMax/454
          RF(NBin)=(LRM*3./(PDense(PrMat)*32.2*4.*PI))**(1./3.)

C         ****************************************************************
C         *** Test for Pennon and End of Run***********************
          IF (HoArea.LT..000000069.AND.Vr.LE.0.00001) THEN
C          IF (HoArea.LT.AC(1)) THEN
           Pennon = .FALSE.
           RETURN
         ELSE IF (Plate.EQ.10.OR.Thick(Plate+1).EQ.0.0) THEN
           Pennon = .TRUE.
           RETURN
         END IF
         ViLRM=Vr
         LastSp=SumSp
         SumSp=SumSp+Space(Plate)
         As=As*(SumSp/LastSp)**2
C        ***********************************************************
C        ************************END OF SHATTER EVALUATION**************
       END IF
2040 CONTINUE
     RETURN
     END
```

```

      SUBROUTINE RFMax(M,RhoP,V,RP,ToverD,MrMax,C,PrMat,Vc,Theta,Kic)
C     M       Initial Impactor Mass                          Grams
C     MrMax   Mass of the Largest Residual Particle          Same as above
C     RhoP    Impactor Density                               Slugs/Ft^3
C     Rp      Initial Impactor Radius (equivalent sphere)    Ft
C     V       Impactor Velocity                              Km/Sec
C     Shock   Velocity                                       Ft/Sec
C     Kic     Fracture Toughness                             MPa m^.5
C     Theta   Approach angle                                 Radians
C     PrMat   Projectile Material 1:Steel, 2:Aluminum ,3:Ice
      REAL Kic,k,MrMax,M,MrOMs,MrOMsC
      REAL MrMaxP,MrOMsP
      REAL MrMaxT,MrOMsT
      REAL RhoP,V,RP,ToverD,C,FTovrD,Vc
      INTEGER PrMat
      k=4.18E6
      IF (ToverD.LT..1) THEN
        FTovrD=1180.
      ELSE IF (ToverD.GE..1.AND.ToverD.LT..2) THEN
        FTovrD=697*ToverD**(-.23)
      ELSE IF (ToverD.GE..2.AND.ToverD.LT..4) THEN
        FTovrD=244*ToverD**(-.881)
      ELSE IF (ToverD.GE..4.AND.ToverD.LT..8) THEN
        FTovrD=1500*ToverD**1.1
      ELSE
        FTovrD=1170.
      END IF
      IF (PrMat.EQ.1) THEN
        FTovrD=FTovrD*7.78
      END IF
      MrOMs = k*(Kic/(RhoP*32.2*C))**2/V**2/(RP*2*12*2.54)*FTovrD
      MrOMsT = MrOMs*COS(Theta)**(-2)
      MrOMsP = MrOMs*COS(Theta)
      Vc =(8.47E4*MrOMs*V**2)**(1/10.93)
      IF (V.LT.Vc) THEN
        MRMaxP=M*MrOMsP
      ELSE
        MrOMsC=1.18E-5*(Vc)**8.93
        MrMaxP=M*MrOMsC*(V/Vc)**(-5.5)
      END IF
      IF (V*COS(Theta).LT.Vc) THEN
        MRMaxT=M*MrOMsT
      ELSE
        MrOMsC=1.18E-5*(Vc)**8.93
        MrMaxT=M*MrOMsC*(V*COS(Theta)/Vc)**(-5.5)
      END IF
      MrMax = AMAX1(MrMaxP,MrMaxT)
      IF (PrMat.EQ.1) MrMax=MrMaxP
      MrMax = AMIN1(MrMax,M*.9999)
      RETURN
      END

      SUBROUTINE PenK(Plate,Thick,NBin,Maxk)
      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk(5),SumPr(5),Pk2
      DOUBLE PRECISION As,Ac,TestAc
      INTEGER Bin,Plate,MaxK(5),I,BinsDo,NBin
      REAL Thick(10),P,Vc
      LOGICAL BINDON(5)
      INTEGER k(5),MinI
      DOUBLE PRECISION kProbs(5,0:15)
      REAL LnPk,LnkFac
      COMMON /Count/kProbs
      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)
      COMMON /Crater/As,Ac(5),P(5)

C     *** k(NBin) is the number of each particle size that must ****
C     *** impact in one crater to make a hole.******************
C     *** This loop finds the fraction of the plate that is not hit by ******
C     *** a fragment of size I (P(0)).  BinDon are initialized***
      TestAc=1D0-1D-3*Ac(1)/As
      BinsDo=0
      DO 10 I=1,NBin
        k(I)=0D0
        CALL Prs(I,Pk(I),k,NBin)
        kProbs(I,0)=Pk(I)
        SumPr(I)=Pk(I)
C       *** If there is not room for one more crater on the plate then ******
C       *** stop using this particle size ***********************************
        IF (SumPr(I).GT.TestAc.OR.
     :         (K(I)*P(I)).GT.Thick(Plate)) THEN
          BinDon(I)=.TRUE.
          BinsDo=BinsDo+1
          kProbs(I,1)=1D0-kProbs(I,0)
        ELSE
          BinDon(I)=.FALSE.
        END IF
C       **************************************************************
  10  CONTINUE
C     END DO
C     ******************************************************************
C     *** This subroutine finds which fragment size has the least SumPr *****
C     *** SumPr is the fraction of the plate that has craters of depth <= k**
C     ******************************************************************
C     *** This loop sums up the area of the plate that is not penetrated,****
C     *** while keeping the area covered by craters of depth <=k for each ***
C     *** size approximately equal.*****************************************
 20   IF (BinsDo.GE.NBin-1) GOTO 30
        CALL MiniI(MinI,BinDon,NBin,SumPr)
        k(MinI) = k(MinI)+1
C       *** This subroutine Calculates the fraction of the plate that is ****
C       *** covered by craters from exactly k particles of size MinI ********
        CALL PrS(MinI,Pk(MinI),k,NBin)
C       **************************************************************
C       *** Add up the fraction of the spray area accounted for, and the ****
C       *** fraction of the particles used so far*************************
        SumPr(MinI)=SumPr(MinI)+Pk(MinI)
        kProbs(MinI,k(MinI))=Pk(MinI)
C       ************************************************************
C       *** If there is not room for one more crater on the plate then ******
C       *** stop using this particle size***********************************
        IF (SumPr(MinI).GT.TestAc.OR.
     :         (K(MinI)*P(MinI)).GT.Thick(Plate)) THEN
          BinDon(MinI)=.TRUE.
          BinsDo=BinsDo+1
          Maxk(MinI)=K(MinI)
        END IF
C       ****************************************************************
      GOTO 20
 30   CONTINUE
       K(NBin) = 1
       kProbs(NBin,1)=1D0-kProbs(NBin,0)
       Maxk(NBin) = 1
C     ******************************************************************
      RETURN
      END

      SUBROUTINE MiniI(MinI,BinDon,NBin,SumPr)
      LOGICAL BinDon(5)
      INTEGER BinsDo
      DOUBLE PRECISION SumPr,MinSum
      DIMENSION SumPr(5)
      MinSum=1D0
      DO 10 I=1,NBin-1
        IF (.NOT.BinDon(I).AND.SumPr(I).LT.MinSum) THEN
          MinI=I
          MinSum=SumPr(I)
        END IF
   10 CONTINUE
C     END DO
      RETURN
      END

      SUBROUTINE PrS(I,Pk,k,NBin)
      DOUBLE PRECISION LastTe(5),PkMin,FPi
      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk,SumPr
      DOUBLE PRECISION As,Ac
      DIMENSION k(5)
      DIMENSION PkMin(5),kMin(5)
      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)
      FPi=1D0/SQRT(2D0*3.1415926D0)
      IF (Nr(I).GE.50.AND.Lambda(I).LT.5) THEN
C       *** This section calculates the poisson approximation to ***
C       *** the binomial distribution*******************************
        IF (K(I).EQ.0)THEN
C         *** Initialize P(k) for first value***********************
          Pk=DEXP(-Lambda(I))
C         **********************************************************
        ELSE
C         *** Calculate P(k) from P(k-1)****************************
          Pk=LastTe(I)*Lambda(I)/K(I)
C         **********************************************************
        END IF
        LastTe(I)=Pk
C       ************************************************************
      ELSE IF(Pcr(I).GT..1.AND.Pcr(I).LT..9.AND.
     :        Nr(I).GT.9./(Pcr(I)*(1-Pcr(I))))THEN
C       *** This section calculates the normal approximation to ****
C       *** the binomial distribution*******************************
        Pk=FPi/Sigma(I)/DEXP(1/(2.D0*SigSq(I)*(k(I)-Lambda(I))**2))
C       ************************************************************
      ELSE
C       ***This section calculates the binomial distribution**********
        IF (k(I).EQ.0) THEN
C         *** Find least non-zero P(k) and corresponding k************
          CALL SetBin(I,PkMin,kMin,NBin)
          LastTe(I)=PkMin(I)
          IF (kMin(I).EQ.0) Pk=PkMin(I)
C         ************************************************************
        ELSE IF (k(I).GT.kMin(I)) THEN
C         *** calculate the non-zero P(k)s****************************
          Pk = LastTe(I)*Pcr(I)/(1D0-Pcr(I))*(Nr(I)-k(I)+1D0)/k(I)
          LastTe(I)=Pk
C         ************************************************************
        ELSE IF (k(I).EQ.KMin(I)) THEN
          Pk = PkMin(I)
        ELSE
          Pk = 0D0
        END IF
C       **************************************************************
      END IF
      RETURN
      END

      SUBROUTINE SetBin(I,PkMin,kMin,NBin)
C     *** The Probability of the mean case occuring is computed ******
C     *** in subroutine Binomial.  P(kMin) is calculated from ********
C     *** this P(Lambda).  The magnitude of P(kMin) is arbitrarily ***
C     *** chosen to be about 1*10^-7.*********************************

      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)

      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk,SumPr,N1,PkMin,P1
      DIMENSION PkMin(5),kMin(5)
      k=Lambda(I)
      N1=Nr(I)
      P1=Pcr(I)
      CALL Binomi(k,N1,Pk,P1)
 1    IF (.NOT.(Pk.GT.1D-13.AND.k.GE.1)) GOTO 10
        Pk = Pk*(1D0-Pcr(I))/Pcr(I)*k/(Nr(I)-k-1D0)
        k = k-1
      GOTO 1
 10   CONTINUE
C     END DO
      kMin(I)=k
      PkMin(I)=Pk
      RETURN
      END
```

```

      SUBROUTINE Binomi (k1,Nr,Pk,Pcr)
      DOUBLE PRECISION Nr
      DOUBLE PRECISION Pk,Pcr,Lambda,LastPk,Qcr
      DOUBLE PRECISION Top,Lower1,Lower2,k,Expon
      Lambda=k1
      k=k1
C     **********Binomial Distribution, Calculates P(k)************
      Lower1 = 0d0
      Lower2 = 0d0
      Top = Nr-k
      Pk = 1d0
      Qcr = 1D0-Pcr
      LastPk = 0D0
      IF (Qcr.EQ.1.D0) THEN
        IF (k.EQ.0.D0) THEN
          Pk=1.D0
        ELSE
          Pk=0.D0
        ENDIF
      ELSE
 3040   IF (Lower1+Lower2+Top.GE.(2*Nr).OR.Pk.EQ.LastPk) GOTO 3030
          LastPk = Pk
 3060     IF (Top.GE.Nr.OR.Pk.GE.1E23) GOTO 3050
            Top = Top + 1D0
            Pk = Pk * Top
          GOTO 3060
 3050     CONTINUE
 3080     IF (Lower1.GE.k.OR.Pk.LE.1E-20) GOTO 3070
            Lower1 = Lower1 + 1d0
            Pk = Pk /Lower1*Pcr
          GOTO 3080
 3070     CONTINUE
          IF (Lower1.GE.k.AND.Lower2.LT.Nr-k.AND.Pk.GT.1D-27) THEN
            Expon=DMIN1(((-28.D0-DLOG10(Pk))/DLOG10(Qcr)),(Nr-k-Lower2))
            IF (Expon.GT.0) Pk=Pk*Qcr**Expon
            Lower2 = Lower2+Expon
          ENDIF
          GOTO 3040
 3030   CONTINUE
      END IF
C     ***********************************************************
      RETURN
      END
```

c-3

```

      SUBROUTINE MasChr (V,M,ToverD,Theta,RP,RhoP,RhoT,C,NBin,
     :                     MenRad,Nr,MrMax,MAvg,PrMat,FrTuff)
      DOUBLE PRECISION NR(5)
      REAL MProj,M,MRMax,MAvg,MenRad(5),MPlate
      REAL MasLim,NrmF,NrmS,BinMen(5),BinMas(0:5)
      INTEGER Bin,PrMat
      PI = 3.14159

C     This subroutine divides the residual mass into bins of equal mass
C     The number of fragments in each bin are also noted

C     ** Convert V to Km/S and MProj to grams
      Vi = V/3281.
      MProj = M * 454.
      MPlate = M * 454.

C     ** Largest Residual Mass **

      IF (PrMat.EQ.3) THEN
        CT = 17569
        RhoT = 5.39
        CALL RFMax(MPlate,RhoT,Vi,RP,ToverD,MRMax,
     :          CT,PrMat,Vc,Theta,FrTuff)
      ELSE
        CALL RFMax(MProj,RhoP,Vi,RP,ToverD,MRMax,
     :          C,PrMat,Vc,Theta,FrTuff)
      END IF

C     ** Average Residual Mass **
      CALL AvgRes (ToverD,Theta,MPlate,Vi,Alfa,MAvg)
      IF (Vi.GT.Vc) MAvg=MAvg*(Vi/Vc)**(-5.5)

C     ** Parameters and Normalization Constants for Weibull Distribution **
      CALL ShCons (ToverD,Theta,MProj,Vi,MAvg,
     :                       bS,sS,bF,sF,NrmS,NrmF,MRMax)

C     ** Size Shatter Begins **
      CALL Shhold (MProj,Vi,TD,Theta,MAvg,bF,sF,bS,sS,
     :                   NrmF,NrmS,FrgLim,MasLim,MRMax)

C     ** Upper Bound and Mean Fragment weight for each Bin **
      CALL BinLim(NBin,MAvg,MRMax,BinMas,BinMen,MProj,
     :               sF,bF,sS,bS,NrmS,NrmF,MasLim)

C     ** Loop Determining Number of Fragments in each Bin in Shatter Regime *
      DO 10 Bin = 1,NBin-1
        Nr(Bin)=(BinMas(Bin)-BinMas(Bin-1))/BinMen(Bin)
  10  CONTINUE
C     END DO
      Nr(NBin) = 1
C     ** Convert Bin Masses into lbs from grams then to Radius in Feet**
      DO 20 Bin = 1,NBin
        BinMen(Bin)=BinMen(Bin)/454.
        MenRad(Bin) = (BinMen(Bin)*3./(RhoP*32.2*4.*PI))**(1./3.)
   20 CONTINUE
C     END DO
      MrMax=MrMax/454.
      MPlate = MPlate/454.
      MProj = MProj/454.
      RETURN
      END
```

```

            SUBROUTINE MasDis (M,MAvg,b,s,NrmC,MasSum)
      C
      C     THIS PROGRAM COMPUTES SHATTER Mass distributions
      C     STEEL CUBES ON ALUMINUM PLATE DATA.
      C          V = IMPACT VELOCITY     (FEET/SECOND)
      C         TD = T OVER D
      C          M = PROJECTILE MASS     (GRAINS)
      C      THETA = ANGLE OF OBLIQUITY
      C       ALPHA=ANGLE OF IMPACT
      C
            REAL MAvg,M,NrmC,MasSum
            W = M/MAvg
      C     **Mass in Bin**
          determ=b+s*LOG(W)
          IF (Determ.LT.-80) THEN
            MasSum=0
          ELSE IF (Determ.LT.4) THEN
            MasSum = NrmC*(1-EXP(-EXP(Determ)))
          ELSE
            MasSum=NrmC
          END IF
          RETURN
          END
```

```
C       SSSSSSSSSSSSSS  SALVO's    CHANGES  SSSSSSSSSSSS

      SUBROUTINE ShCons(TD,Theta,M,Vi,MAvg,
     :                            bS,sS,bF,sF,NrmCS,NrmCF,LRM)
      REAL M,NrmCS,NrmCF,MT2,MAvg,IntegS,IntegF,LRM
          sS=1.8-.04*TD/COS(THETA)-.042*M
     &        +.34*(COS(2.*ALPHA)**2.)-(1-EXP(-.64*Vi))
          bS=-2.3-1.1*TD/COS(THETA)+.0675*M
     &        -.27*Vi+1.4*(COS(2.*ALPHA)**2.)
          sF=1.38-.510*TD/COS(THETA)+.036*M
     &        +3.31*(COS(2.*ALPHA)**2.)-(1-EXP(-.390*Vi))
          bF=-1.17+.313*TD/COS(THETA)+.0675*M
     &        +.508*Vi-1.41*(COS(2.*ALPHA)**2.)

C     ** Total mass in shatter regime **
      MT2 =(-.957+EXP(-.0013*M))*(Vi**(.38*M+2.5))
      MT2 = AMAX1(MT2,1E-20)
      MT2 = AMIN1(MT2,M)
C       **Normalization to Largest Residual Mass**
C        IntegS = 1-EXP(-EXP(bS)*(M/MAvg)**sS)
C        IntegF = 1-EXP(-EXP(bF)*(M/MAvg)**sF)
C        NrmCS = MT2/IntegS
C        NrmCF = M  /IntegF
      detrm1=bS+sS*ALOG(LRM/MAvg)
      IF (Detrm1.LT.-15) THEN
        NrmCS=1e-30
      ELSE IF (Detrm1.LT.4) THEN
        IntegS = 1-EXP(-EXP(Detrm1))
        NrmCS = MT2/IntegS
      ELSE
        NrmCS = MT2
      END IF
      detrm2=bF+sF*LOG(LRM/MAvg)
      IF (Detrm2.LT.-15) THEN
        NrmCS=1e-30
      ELSE IF (Detrm2.LT.4) THEN
        IntegF = 1-EXP(-EXP(Detrm2))
        NrmCF = M/IntegF
      ELSE
        NrmCF = M
      END IF
        if(sF)1,2,2
1       sF=0.
2       if(sS)3,4,4
3       sS=0.
4     RETURN
      END
      SUBROUTINE Shhold (MProj,Vi,TD,Theta,MAvg,bF,sF,bS,sS,
     :                          NrmF,NrmS,FrgLim,MasLim,LRM)
      REAL LowLim,Mean,Mfs,Mss,MAvg,MasLim,MProj,NrmF,NrmS,M,LRM
      HiLim = MProj
      LowLim = 0
      DO 10 I=1,20
        Mean = (HiLim+LowLim)/2
        CALL MasDis (Mean,MAvg,bF,sF,NrmF,Mfs)
        CALL MasDis (Mean,MAvg,bS,sS,NrmS,Mss)
        IF (Mfs.GT.Mss) THEN
          HiLim = Mean
        ELSE
          LowLim = Mean
        END IF
   10 CONTINUE
C     END DO
      FrgLim = Mean
      MasLim = AMIN1(Mss,MProj-LRM)
      RETURN
      END
      SUBROUTINE BinLim (NBin,MAvg,LRM,BinMas,BinMen,MProj,
     :                       sF,bF,sS,bS,NrmS,NrmF,MasLim)
      REAL MAvg,LRM,MProj,MasLim,NrmS,NrmF,BinInt,MasMen,MFrac(0:4)
      DIMENSION BinMen(5),BinMas(0:5),MasMen(5)
      INTEGER Bin,ShtBin
      DATA MFrac/0.,.25,.5,.75,1./
      ShtBin=0
      DO 5 Bin=1,NBin-1
        BinMas(Bin) = (MProj-LRM)*MFrac(Bin)
        MasMen(Bin) = (MProj-LRM)*(MFrac(Bin)+MFrac(Bin-1))/2.
        IF (BinMas(Bin).LE.MasLim) ShtBin=Bin
    5 CONTINUE
C     END DO
C     ** Bin Limit and Mean for Shatter **
      DO 10 Bin = 1,ShtBin
       BinMen(Bin) = MAvg*(-EXP(-bS)*DLOG(1D0-MasMen(Bin)/NrmS))**(1/sS)
   10 CONTINUE
C     END DO
C     ** Bin Limit and Mean for First Bin in Fracture **
      Bin = ShtBin+1
      IF (Bin.LE.NBin-1) THEN
        IF (MasLim.GE.MasMen(Bin)) THEN
C         ** Bin Mean in Case it is Less Than the Fracture Threshold **
         BinMen(Bin)=MAvg*(-EXP(-bS)*DLOG(1D0-MasMen(Bin)/NrmS))**(1/sS)
        ELSE
         BinMen(Bin)=MAvg*(-EXP(-bF)*DLOG(1D0-MasMen(Bin)/NrmF))**(1/sF)
        END IF
      END IF
C     ** Bin Limit and Mean for Fracture **
      DO 20 Bin = ShtBin+2,NBin-1
       BinMen(Bin) = MAvg*(-EXP(-bF)*DLOG(1D0-MasMen(Bin)/NrmF))**(1/sF)
   20 CONTINUE
C     END DO
C     ** Bin Mean for largest bin is LRM (Largest Residual Mass) **
      BinMen(NBin) = LRM
      RETURN
      END
      SUBROUTINE AvgRes (ToverD,Theta,M,V,Alfa,MAvg)
      REAL MAvg,M
C     ** Average Fragment Mass **
      MAvg = .0109-.00879*ToverD/COS(Theta)+.000506*M-.00428*V+
     :        .0110*COS(2*Alfa)**2
      MAvg = AMax1(MAvg,0.005)
      RETURN
      END

      SUBROUTINE Countr(NBin,P,Thick,Maxk,Intact,Nr,Ac,As)
      INTEGER Bin,NBin,SmBin
      INTEGER Digit(5),HoMin(5),I,Maxk(5)
      REAL Thick,HoDept,P(5)
      DOUBLE PRECISION Probs(5,0:15)
      DOUBLE PRECISION Intact,ITArea,ITArS1
      DOUBLE PRECISION Nr(5),Ac(5),As
      DOUBLE PRECISION ArHol1,ArHol2,ArHol3
      DOUBLE PRECISION ArHol4,ArHol5
      DOUBLE PRECISION PrTemp,DigTp1,DigTp2,DigTp3
      DOUBLE PRECISION           DigTp4,DigTp5
      LOGICAL UnInc,OBin1D
      COMMON /Count/Probs
      DO 10 Bin = 1,NBin
        HoMin(Bin) = JMIN0(INT(Thick/P(Bin)+1),Maxk(Bin))
        Digit(Bin)=0
 10   CONTINUE
C     END DO
      ArHol1=0.D0
      ArHol2=0.D0
      ArHol3=0.D0
      ArHol4=0.D0
      ArHol5=0.D0
      Intact=0.D0
      Bin = 0
      HoDept = 0
      OBinld = .FALSE.
C     DO WHILE (Bin.LE.NBin)
 2    IF (.NOT.(Bin.LE.NBin)) GOTO 20
        IF (HoDept.GT.Thick) THEN
          OBinld = .FALSE.
          SmBin=1
C         DO WHILE (Digit(SmBin).EQ.0)
 21       IF (.NOT.(Digit(SmBin).EQ.0)) GOTO 210
            SmBin=SmBin+1
          GOTO 21
 210      CONTINUE
C         END DO
          Digit(SmBin)=0
          HoDept=0.
          Bin = SmBin+1
          UnInc = .TRUE.
C         DO WHILE (Bin.LE.NBin.AND.UnInc)
 22       IF (.NOT.(Bin.LE.NBin.AND.UnInc)) GOTO 220
            IF (Digit(Bin).LT.HoMin(Bin)) THEN
              Digit(Bin) = Digit(Bin)+1
              UnInc = .FALSE.
            ELSE
              Digit(Bin)=0
              Bin = Bin + 1
            END IF
          GOTO 22
 220      CONTINUE
C         END DO
          DO 230 I=SmBin+1,NBin
            HoDept=HoDept+Digit(I)*P(I)
 230      CONTINUE
C         END DO
        ELSE
          IF (OBinld) THEN
            DigTp1=DigTp1+1
            ITArea = Probs(1,DigTp1)*ITArS1
            ArHol1 = ArHol1+ITArea*DigTp1
            ArHol2 = ArHol2+ITArea*DigTp2
            ArHol3 = ArHol3+ITArea*DigTp3
            ArHol4 = ArHol4+ITArea*DigTp4
            ArHol5 = ArHol5+ITArea*DigTp5

          ELSE
            ITArea=1.D0

            DigTp2=Digit(2)
            PrTemp = Probs(2,DigTp2)
            ITArea = PrTemp*ITArea

            DigTp3=Digit(3)
            PrTemp = Probs(3,DigTp3)
            ITArea = PrTemp *ITArea

            DigTp4=Digit(4)
            PrTemp = Probs(4,DigTp4)
            ITArea = PrTemp *ITArea

            DigTp5=Digit(5)
            PrTemp = Probs(5,DigTp5)
            ITArea = PrTemp *ITArea

            ITArS1=ITArea

            DigTp1=Digit(1)
            PrTemp = Probs(1,DigTp1)
            ITArea = PrTemp *ITArea

            ArHol1 = ArHol1+ITArea*DigTp1
            ArHol2 = ArHol2+ITArea*DigTp2
            ArHol3 = ArHol3+ITArea*DigTp3
            ArHol4 = ArHol4+ITArea*DigTp4
            ArHol5 = ArHol5+ITArea*DigTp5

          END IF
          Intact=Intact+ITArea
          Bin = 1
          OBinld = .TRUE.
          UnInc = .TRUE.
C         DO WHILE (Bin.LE.NBin.AND.UnInc)
 24       IF (.NOT.(Bin.LE.NBin.AND.UnInc)) GOTO 240
            IF (Digit(Bin).LT.HoMin(Bin)) THEN
              Digit(Bin) = Digit(Bin)+1
              HoDept = HoDept+P(Bin)
              UnInc = .FALSE.
            ELSE
              OBinld = .FALSE.
              HoDept = HoDept-Digit(Bin)*P(Bin)
              Digit(Bin)=0
              Bin = Bin + 1
            END IF
          GOTO 24
 240      CONTINUE
C         END DO
        END IF
      GOTO 2
 20   CONTINUE
C     END DO
        Nr(1)=DMAX1(0D0,Nr(1)-ArHol1*As/Ac(1))
```

```
     Nr(2)=DMAX1(0D0,Nr(2)-ArHol2*As/Ac(2))
     Nr(3)=DMAX1(0D0,Nr(3)-ArHol3*As/Ac(3))
     Nr(4)=DMAX1(0D0,Nr(4)-ArHol4*As/Ac(4))
     Nr(5)=DMAX1(0D0,Nr(5)-ArHol5*As/Ac(5))
   RETURN
   END
```

```
      SUBROUTINE ResVel
     :   (Vr,V0,RP,RhoP,RhoT,Thick,Theta,PrMat,TarMat)
C        This is the JTCG Residual Velocity Formula
      INTEGER PrMat,TarMat
      PresAr = 3.14159*RP**2
      V0cm = V0/.03281
      Weight = 4./3.*3.14159*RP**3*RhoP
      V50 = BalLim(RP,RhoP,RhoT,Thick,Theta,PrMat,TarMat)
      Q4 = RhoT*Thick*PresAr/(Weight*COS(Theta))
      VrSq = AMAX1(0.,V0cm**2-V50**2)
      Vr = SQRT(VrSq) / (1.+Q4)
      Vr = Vr*.03281
      RETURN
      END

      FUNCTION BalLim
     :                    (RP,RhoP,RhoT,Thick,Theta,PrMat,TarMat)
      INTEGER PrMat,TarMat
      IF (PrMat.EQ.3) THEN
        BalLim = 2.45*3281*(Thick/12./2.54)**(-.15)*
     :      (Thick/RP/2.)*(RhoT/RhoP)**.64*(1./COS(Theta))**1.01
      ELSE
C       This is the JTCG V50 formula
        PresAr = 3.14159*RP**2
        Weight = 4./3.*3.14159*RP**3*RhoP*32.2
        W0 = .0143
        IF (PrMat.EQ.1) THEN
          IF (TarMat.EQ.2) THEN
            Cbf = 41300.
            Bf  =  .941
            H   = 1.098
            F   = -.038
          ELSE
            Cbf = 80600.
            Bf  =  .963
            H   = 1.286
            F   = -.057
          END IF
        ELSE
          Cbf = 92800
          Bf  =  .972
          H   = 1.01
          F   = -.105
        END IF
        Q8 = RhoP*32.2*Thick*PresAr/Weight
        Q11 = RhoP*32.2*Thick*PresAr/W0
        BalLim = Cbf*Q8**Bf/COS(Theta)**H*Q11**F
      END IF
      RETURN
      END
```

FILE - DEB.VEL

```
 1  0.00   3.21  1.054336E-04  1.368229E-03  7.253182E-04  2.749629E-04
 2  3.21   4.00 -4.229632      3.795033     -1.134365      1.135913E-01
 3  4.00   4.11 -4.200628E+03  3.110938E+03 -7.679830E+02  6.319804E+01
 4  4.11   5.16 -1.622625      7.572647E-01 -8.053452E-02 -4.511672E-04
 5  5.16   5.96  2.043598E+01 -1.042294E+01  1.775327     -1.007870E-01
 6  5.96   7.18  7.860538     -3.547154      5.318564E-01 -2.631684E-02
 7  7.18   8.10 -1.915112E-01 -1.182071E-01  4.134483E-02 -2.761015E-03
 8  8.10   8.67  6.661122E+01 -2.432737E+01  2.967511     -1.207315E-01
 9  8.67   9.97 -3.037310E+01  1.024428E+01 -1.147075      4.273907E-02
10  9.97  10.25 -4.609121      4.719100E-01  0.0           0.0
11 10.25  10.72 -3.250999E+02  9.131992E+01 -8.543601      2.664182E-01
12 10.72  11.42  3.257151E+02 -8.911450E+01  8.133017     -2.474180E-01
13 11.42  11.56  3.994901     -3.317631E-01  0.0           0.0
14 11.56  12.36  1.701649E+02 -4.206065E+01  3.470595     -9.552701E-02
15 12.36  13.61  4.714685E+01 -1.113553E+01  8.805859E-01 -2.325908E-02
16 13.61  17.00  1.578160E+01 -2.934914      1.820184E-01 -3.763893E-03
```

APPENDIX D

Source Code for BUMPER Module

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                   C
C      BUMPER VER 4.0  5/25/87      C
C                                   C
C       BOEING AEROSPACE CO.        C
C                                   C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C  Bumper Ver 4.0 predicts the probability of no penetration for
C  spacecraft subject to man-made orbital debris or meteoroid impact.
C  The spacecraft is assumed to be operating in low earth orbit ( approx.
C  500km ). The code accounts for varying impact velocity, impact angle,
C  wall configurations, and the effects of spacecraft geometry and
C  orientation. It is currently limited to the case of spheres
C  ( debris or meteroids ) impacting conventional aluminum two plate
C  structures, with or without multi-layer insulation between the plates.
C
C  The code requires 2 files generated by other codes. The first is the
C  output file from the GEOMETRY code. This file contains the threat
C  information and the element id, pid, and surface area lists. In
C  addition the code outputs a list of the exposed elements and their
C  impact angles for each threat case.
C
C  The second file is the output file from the RESPONSE code. This file
C  contains the critical diameter tables for each wall configuration as
C  a function of impact velocity and impact angle.
C
C
C  The code also produces a Supertab universal file which may be used in
C  conjuction with the orginal Supertab model file to graphically
C  display the distribution of the probability of penetration on the
C  spacecraft.
C
C  The code was developed under the NASA contract 'Integrated Wall Design
C  Guide and Penetration Control Plan' by M.A.Wright & A.R.Coronado
C
C
C  Common Block Variable list
C
C  Scalers
C
C    alt = operating altitude , km
C    binc = impact angle (beta) increment , deg
C    cbeta = cosine of beta
C    diam = critical diameter , cm
C    etime = spacecraft exposure time , years
C    flx = number of impacts per projected area per year of diameter D
C           or larger
C    it = current threat case
C    itype = analysis type , 1- debris, 2-meteoroids
C    nb = number of angles in the response array
C    nc = number of wall configurations in the response array
C    nel = current element number
C    nelm = total number of elements
C    nr = number of element ranges to sum over
C    nt = number of threat cases
C    nv = number of velocities in the response array
C    vr = impact (relative) velocity , km/sec
C    vinc = impact (relative) velocity increment , km/sec
C
```

```
C Arrays
C
C   area - array containing the value of the surface area for each
C          element, sq-meters
C   exposed - list of the number of exposed elements for each threat
C             angle.
C   geometry - array containing the values of the cosine of the impact
C              angle for each exposed element for each threat angle.
C   id - array containing the values of the element and property id
C        for each element
C        1- id
C        2- pid
C   point - array of the element numbers corresponding to the elements
C           in the geometry array.
C   range - array containing the starting and ending elment id for each
C           range to sum over
C           1-starting id
C           2- ending id
C   response - array containing the values of the critical diameter as
C              a function of impact angle and velocity.
C              (vr,beta,pid)
C   threat - array contaning threat information
C            1-theta angle, rad
C            2-phi angle, rad
C            3-vr, km/sec
C            4-prob
C
C
C Main Program Variable List
C
C Scalers
C
C   answer - user input
C   ae - effective area term, product of the cosine of the impact
C        angle and the probability of the threat occuring
C   fa - product of the threat probability, flux and cosine(beta),
C        impacts/yr/sq-meter
C   prob - probabiltiy of threat case i occuring
C   taeff - total effective area, sq-m
C   tpnp - total probability of no penetration for all elements, %
C   tsum - total sum of the history array
C
C Arrays
C
C   aeff - array containing the effective area for reach range
C   arhis - array containing the running sum of the effective area
C           term for each element
C   history - list containing the  running sum of the fa term for each
C             element
C   pnp - array containing the probability of no penetration for each
C         range
C   sum - array contaning the sum of the history array for each element
C         range
C
C
      CHARACTER*80 ANSWER
C
      INCLUDE 'COMMON2.BLK'
C
      REAL*8 AE,FA,PROB,TAEFF,TPNP,TSUM,
     1       ARHIS(IEL),HISTORY(IEL),AEFF(50),PNP(50),SUM(50)
C
C Write header to screen, read in screen inputs
```

```
C
      CALL INPUT
C
C  Read in the GEOMETRY output file
C
      CALL GEOREAD
C
C  Read in the RESPONSE output file
C
      CALL RESREAD
C
C  Initialize Tsum and Taeff to 0.0
C
      TAEFF=0.0D0
      TSUM=0.0D0
C
C  Initialize History to 0.0
C
      DO 50 I=1,NELM
         HISTORY(I)=0.0D0
   50 CONTINUE
C
C  Initialize Sum, Aeff, and Pnp to 0.0
C
      DO 70 I=1,50
         AEFF(I)=0.0D0
         PNP(I)=0.0D0
         SUM(I)=0.0D0
   70 CONTINUE
C
C  Determine the penetrating flux for each element, for each threat
C  angle
C
      DO 200 I=1,NT
C
C  Set the threat index and get the impact velocity and the threat
C  probability from the threat array
C
         IT=I
         VR=THREAT(3,IT)
         PROB=THREAT(4,IT)
C
C  Evaluate each exposed element
C
         DO 100 J=1,EXPOSED(I)
C
C  Set the element number
C
            NEL=POINT(J,I)
C
C  Get the cosine of the impact angle from the Geometry array.
C
            CBETA=GEOMETRY(J,I)
C
C  Determine the diameter of the sphere that just penetrates the wall
C
            CALL CRITDIA
C
C  Calculate the flux of the critical diameter using the appropriate
C  flux equation based on the analysis type.
C
            CALL FLUX
C
```

```
C  Multiply the flux by the probability of the threat angle and the
C  cosine of the impact angle, this determines the number of
C  penetrations per year per element surface area for the current
C  element and the current threat angle.
C
            FA=FLX*PROB*CBETA
C
C  Store the running sum of the FA term for each element in the History
C  array. This value represents the total number of penetrations for
C  a given element per year per element surface area.
C
            HISTORY(NEL)=HISTORY(NEL)+FA
C
C  Calculate the effective area term
C
            AE=PROB*CBETA
C
C  Store the running sum of the effective area term in the arhis array
C
            ARHIS(NEL)=ARHIS(NEL)+AE
C
C  Next element
C
  100    CONTINUE
C
C  Write message to the screen indicating current threat has been
C  evaluated
C
         WRITE ( 6,150 ) IT
  150    FORMAT ( 1X,'THREAT CASE ',I4,1X,'COMPLETED' )
C
C  Next threat
C
  200 CONTINUE
C
C  Multiply each term in the HISTORY and ARHIS arrays by the appropriate
C  element surface area.
C
      DO 250 I=1,NELM
         HISTORY(I)=HISTORY(I)*AREA(I)
         ARHIS(I)=ARHIS(I)*AREA(I)
  250 CONTINUE
C
C  Sum up the HISTORY and ARHIS arrays by components.
C
       IC=1
C
       DO 310 I=1,NELM
C
          TAEFF=TAEFF+ARHIS(I)
          TSUM=TSUM+HISTORY(I)
C
  300     IS=RANGE(1,IC)
          IE=RANGE(2,IC)
C
          IF ( IC.GT.NR ) GO TO 310
          IF ( IS.GT.ID(1,I) ) GO TO 310
          IF ( IE.LT.ID(1,I) ) THEN
             IC=IC+1
             GO TO 300
          END IF
C
          AEFF(IC)=AEFF(IC)+ARHIS(I)
```

```
               SUM(IC)=SUM(IC)+HISTORY(I)
C
  310 CONTINUE
C
C  Calculate the probabilty of no penetration for each range using
C  a Possion model
C
      DO 320 I=1,NR
         PNP(I)=(DEXP(-SUM(I)*ETIME))*100.D0
  320 CONTINUE
C
C  Calculate the total PNP
C
      TPNP=(DEXP(-TSUM*ETIME))*100.0D0
C
C  Write the probability value to the screen and the summary file.
C
      WRITE ( 6,390 )
      WRITE ( 10,390 )
  390 FORMAT ( /1X,'RANGE',2X,'STARTING EID',2X,'ENDING EID',3X,
     1         'PNP %',6X,'Aeff sq-m' )
      DO 400 I=1,NR
         WRITE ( 6,420 )I,RANGE(1,I),RANGE(2,I),PNP(I),AEFF(I)
         WRITE ( 10,420 )I,RANGE(1,I),RANGE(2,I),PNP(I),AEFF(I)
  400 CONTINUE
  420 FORMAT ( 2X,I4,5X,I8,3X,I8,3X,F10.5,2X,F10.5 )
C
C  Write out totals to screen and save file
C
      WRITE ( 6,430 ) TPNP
      WRITE ( 10,430 ) TPNP
  430 FORMAT ( /1X,'TOTAL PRROBABILITY OF NO PENETRATION (%) = ',F12.5 )
C
      WRITE ( 6,440 ) TAEFF
      WRITE ( 10,440 ) TAEFF
  440 FORMAT ( /1X,'TOTAL EFFECTIVE AREA (SQ-M) = ',F12.5 )
C
C
C  Determine if a Supertab output file is to be created
C
      WRITE ( 6,500 )
  500 FORMAT (/1X,'CREATE A SUPERTAB INPUT FILE FOR CONTOUR PLOTS ?',
     1         1X,'<CR=YES> >',$)
C
      READ ( 5,510 )ANSWER
  510 FORMAT ( A )
C
      IF ( ANSWER(1:1).EQ.'Y' .OR. ANSWER(1:1).EQ.' ' ) THEN
         CALL SUPER (HISTORY)
      END IF
C
C  Close summary file
C
      CLOSE ( UNIT=10,STATUS='KEEP' )
C
C  Finished
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE INPUT
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Input writes the program header to the screen and reads in the
C  summary ouput filename. It also determines the analysis type, the
C  spacecraft exposure time, operating altitude and the element id sum
C  ranges.
C
C
C
C  note: for variables contained in the common block refer to the main
C        listing for definition
C
C  Variable list
C
C    answer = character string representing user input
C    sfile = summary output filenanme
C
C
C
      INCLUDE 'COMMON2.BLK'
C
      CHARACTER*80 ANSWER,SFILE
C
C  Write header to screen and summary file
C
      WRITE ( 6,10 )
   10 FORMAT (/1X,'***********************',//1X,3X,'BUMPER VER 4.0',
     1        //1X,'Last Update 5/25/87',//1X,'***********************')
C
C  Read in summary output filename, set default to sum.dat
C
   15 WRITE ( 6,20 )
   20 FORMAT ( /1X,'SUMMARY OUTPUT FILENAME (CR=BUMPER.SUM)>',$)
      READ ( 5,30 )SFILE
   30 FORMAT (A)
C
      IF ( SFILE(1:1).EQ.' ' ) SFILE='BUMPER.SUM'
C
C  Open sfile
C
      OPEN ( UNIT=10,FILE=SFILE,STATUS='NEW',IOSTAT=IER,ERR=40 )
C
      GO TO 70
C
C  Error control
C
   40 IF ( IER.EQ.2013 ) THEN
         WRITE ( 6,50 )
   50    FORMAT ( /1X,'FILE ALREADY EXISTS OK TO OVERWRITE (CR=YES,$)>')
         READ ( 5,30 ) ANSWER
C
         IF ( ANSWER(1:1).EQ.'Y' .OR. ANSWER(1:1).EQ.' ' ) THEN
            OPEN ( UNIT=10,FILE=SFILE,STATUS='UNKNOWN',IOSTAT=IER,
     1             ERR=40)
            REWIND 10
         ELSE
            GO TO 15
            END IF
         ELSE
            WRITE ( 6,60 )
   60       FORMAT (/1X,'UNABLE TO OPEN FILE ' )
            GO TO  15
         END IF
C
C  Determine analysis type, set default to 1 (debris)
C
   70 WRITE ( 6,80 )
   80 FORMAT (/1X,'ANALYSIS TYPE ?',/,2X,'1-DEBRIS <CR> ',/,2X,
     1        '2-METEOROIDS',/,1X,'ANSWER 1 OR 2 >',$)
C
         READ ( 5,30 ) ANSWER
C
         IF ( ANSWER(1:1).EQ.' ' ) THEN
            ITYPE=1
         ELSE
            READ ( ANSWER(1:80),90,ERR=70 )ITYPE
   90       FORMAT ( BN,I4 )
         END IF
C
C  Check that input was correct
C
         IF ( ITYPE.EQ.1 .OR. ITYPE.EQ.2 ) THEN
            CONTINUE
         ELSE
            WRITE ( 6,100 )
  100       FORMAT ( /1X,'INCORRECT INPUT' )
            GO TO 70
         END IF
C
C  Determine the spacecraft exposure time , set default to 10 years
C
  105 WRITE ( 6,110 )
  110 FORMAT (/1X,'SPACE STATION EXPOSURE TIME (YEARS) <CR=10.0> : ',$)
         READ ( 5,30 ) ANSWER
C
         IF ( ANSWER(1:1).EQ.' ' ) ANSWER='10.0'
C
         READ ( ANSWER(1:80),120,ERR=105 ) ETIME
  120 FORMAT ( BN,D20.0 )
C
C  Write analysis type and etime to summary file
C
         WRITE ( 10,10 )
C
         IF ( ITYPE.EQ.1 ) THEN
            WRITE ( 10,130 )
  130       FORMAT ( /1X,'MAN-MADE ORBITAL DEBRIS ANALYSIS')
         ELSE
            WRITE ( 10,140 )
  140       FORMAT ( 1X,'METEOROID ANALYSIS' )
         END IF
C
         WRITE ( 10,150 )ETIME
  150 FORMAT ( 1X,'SPACECRAFT EXPOSURE TIME (YEARS) =',F7.2 )
C
C  Read in operating altitude , set default to 500 km
C
  160 WRITE ( 6,170 )
  170 FORMAT ( /1X,'OPERATING ALTITUDE ( 400-500 km )  <CR=500> : ',$)
         READ ( 5,30 ) ANSWER
```

```
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         ALT=500.0D0
      ELSE
         READ ( ANSWER(1:80),180,ERR=160 ) ALT
  180    FORMAT ( BN,D20.0 )
      END IF
C
C  Check that altitude is within range
C
      IF ( ALT.LT.350.0 .OR. ALT.GT.550.0 ) THEN
         WRITE ( 6,190 )
  190    FORMAT ( 1X,'---ERROR--- Altitude outside of range' )
         GO TO 160
      END IF
C
C  Write altitude to output file
C
      WRITE ( 10,200 ) ALT
  200 FORMAT ( 1X,'OPERATING ALTITUDE (km) = ',F7.2 )
C
C  Read in element ranges to sum over
C
      IC=0
C
      WRITE ( 6,250 )
  250 FORMAT ( /1X,'THE PROBABILITY OF NO PENETRATION WILL BE ',
     1         'CALCULATED FOR SPECIFIC RANGES',/1X,'OF ELEMENT IDS ',
     2         'INPUT THE STARTING AND ENDING ELEMENT ID FOR ',
     3         'EACH RANGE',/1X,'ENTER D <CR> OR <CR> WHEN DONE')
C
  270 IC=IC+1
  275 WRITE ( 6,280 )IC
  280 FORMAT ( /1X,'RANGE',I4 )
C
  285 WRITE ( 6,290 )
  290 FORMAT ( 1X,'STARTING ELEMENT ID : ',$)
      READ ( 5,30 ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' .OR. ANSWER(1:1).EQ.'D' ) GO TO 500
C
      READ ( ANSWER(1:80),300,ERR=285 ) RANGE(1,IC)
  300 FORMAT ( BN,I12 )
C
  305 WRITE ( 6,310 )
  310 FORMAT ( 1X,'ENDING ELEMENT ID   : ',$)
      READ ( 5,30 ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' .OR. ANSWER(1:1).EQ.'D' ) GO TO 305
C
      READ ( ANSWER(1:80),300,ERR=305 ) RANGE(2,IC)
C
C Check that ending id > starting id
C
      IF ( RANGE(1,IC).GT.RANGE(2,IC) ) THEN
         WRITE ( 6,320 )
  320    FORMAT ( 1X,'---ERROR--- Staring ID greater then Ending ID')
         GO TO 275
      END IF
C
C  Next Range
C
      GO TO  270
```

```
C
C  Check that values were input
C
  500 CONTINUE
      IF ( RANGE(1,1).EQ.0.0 .AND. RANGE(2,1).EQ.0.0 ) THEN
         WRITE ( 6,330 )
  330    FORMAT ( 1X,'---ERROR--- No Range Values Input' )
         IC=0
         GO TO 270
      END IF
C
C  Set the number of ranges equal to the number read in
C
      NR=IC-1
C
C  Finished
C
C     RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE GEOREAD
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Georead reads in the output file from the GEOMETRY code. This file
C  contains the global threat and element data as well as the list of
C  exposed elements and their impact angles for each threat case.
C
C
C
C  note: for variables contained in the common block refer to main
C        listing for definition
C
C
C  Variable List
C
C    answer = character string representing user input
C    gfile = geometry output filename
C    itf = analysis type contained in the
C
C
      INCLUDE 'COMMON2.BLK'
C
      CHARACTER*80 ANSWER,GFILE
C
      INTEGER*2 ITF
C
C  Read in the GEOMETRY output filename, set the default to geom.dat
C
   10 WRITE ( 6,20 )
   20 FORMAT(/1X,'GEOMETRY OUTPUT FILENAME ? <CR=STATION.GEM> >',$)
      READ ( 5,30 ) GFILE
   30 FORMAT (A)
C
      IF ( GFILE(1:1).EQ.' ' ) GFILE='STATION.GEM'
C
C  Open the file
C
      OPEN (UNIT=2,FILE=GFILE,STATUS='OLD',FORM='UNFORMATTED',ERR=40 )
C
      GO TO  60
C
C  Error control
C
   40 WRITE ( 6,50 )
   50 FORMAT ( /1X,'UNABLE TO OPEN FILE ' )
       GO TO 10
C
C  Read in the analysis type,the number of threat cases, and the
C  number of elements
C
   60 READ (2) ITF,NT,NELM
C
C  Check that the analysis type in the file matches the type input
C  by user.
C
      IF ( ITF.NE.ITYPE ) THEN
         IF ( ITYPE.EQ.1 ) THEN
            WRITE ( 6,70 )
   70       FORMAT ( /1X,'DEBRIS ANALYSIS SPECIFIED BUT FILE IS FOR ',
     1               'METEOROIDS')
         ELSE
            WRITE ( 6,80 )
   80       FORMAT ( /1X,'METEOROID ANALYSIS SPECIFIED BUT FILE IS ',
     1               'FOR DEBRIS' )
         END IF
C
         WRITE ( 6,90 )
   90    FORMAT ( /1X,'DO YOU WISH TO CONTINUE (CR=NO) >',$)
         READ ( 5,30 ) ANSWER
C
         IF ( ANSWER(1:1).EQ.'Y' ) THEN
            GO TO 10
         ELSE
            STOP
         END IF
C
      END IF
C
C   Check that the number of threats and the number of elements are less
C   then the maximum allowed
C
      IF ( NT.GT.ITH ) THEN
         WRITE ( 6,100 )
  100    FORMAT ( /1X,'NUMBER OF THREATS IS GREATER THEN MAX ALLOWED')
         WRITE ( 6,105 )
  105    FORMAT ( 1X,'ARRAY SIZE MUST BE INCREASED & CODE RECOMPILED')
         STOP
      END IF
C
      IF ( NELM.GT.IEL ) THEN
         WRITE ( 6,110 )
  110    FORMAT ( /1X,'NUMBER OF ELEMENTS IS GREATER THEN MAX ALLOWED')
         WRITE ( 6,105 )
         STOP
      END IF
C
C  Initialize the arrays to 0.0
C
      DO 150 I=1,NT
         THREAT(3,I)=0.0
         THREAT(4,I)=0.0
         EXPOSED(I)=0
         DO 140 J=1,NELM
            GEOMETRY(J,I)=0.0
            ID(1,J)=0
            ID(2,J)=0
            POINT(J,I)=0
  140    CONTINUE
  150 CONTINUE
C
C  Read in the Threat data
C
      DO 175 I=1,NT
         READ (2) (THREAT(J,I),J=1,4)
  175 CONTINUE
C
C  Read in the element id, and  property id storing them in the ID
C  array.
C
      DO 200 I=1,NELM
         READ (2) (ID(J,I),J=1,2)
  200 CONTINUE
C
C  Read in the element's surface area storing it in the AREA array.
C
      DO 250 I=1,NELM
         READ (2) AREA(I)
  250 CONTINUE
C
C  Read in the geometry data for the exposed elements
C
      DO 500 I=1,NT
C
C  Read in the threat case and the number of exposed elements
C
         READ (2) IT,EXPOSED(I)
C
C  Loop thru the exposed elements
C
         DO 400 J=1,EXPOSED(I)
C
C  Read in the element number (storing in the POINT array), and the
C  cosine of the impact angle (storing in the GEOMETRY array).
C
            READ (2) POINT(J,I),GEOMETRY(J,I)
C
  400    CONTINUE
C
  500 CONTINUE
C
C  Write gfile to summary file
C
      WRITE ( 10,600 )GFILE
  600 FORMAT ( 1X,'GEOMETRY OUTPUT FILE = ',A )
C
C  Close file
C
      CLOSE ( UNIT=2,STATUS='KEEP' )
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE RESREAD
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Resread reads in the output from the RESPONSE code.  This output
C  consists of the critical diameter data as a function of property
C  id, impact angle, and impact velocity.
C
C
C  note: for variables contained in the common block refer to the main
C        listing for definition.
C
C
C  Variable list
C
C    answer = character string represnting user input
C    itf = analysis type for rfile
C    rfile = response output filename
C
C
      CHARACTER*80 ANSWER,RFILE
C
      INTEGER*2 ITF
C
      INCLUDE 'COMMON2.BLK'
C
C  Read in the RESPONSE output filename , set default to resp.dat
C
   10 WRITE ( 6,20 )
   20 FORMAT(/1X,'RESPONSE OUTPUT FILENAME ? <CR=STATION.RSP> >',$)
      READ ( 5,30 ) RFILE
   30 FORMAT (A)
C
      IF ( RFILE(1:1).EQ.' ' ) RFILE='STATION.RSP'
C
C  Open the file
C
      OPEN ( UNIT=2,FILE=RFILE,STATUS='OLD',FORM='UNFORMATTED',ERR=40 )
C
      GO TO 60
C
C  Error control on open
C
   40 WRITE ( 6,50 )
   50 FORMAT ( /1X,'UNABLE TO OPEN FILE' )
      GO TO 10
C
C  Read in the analysis type and the number of property cases.
C
   60 READ (2) ITF,NC
C
C  Check that the response file is the correct analysis type
C
        IF ( ITF.NE.ITYPE ) THEN
          IF ( ITYPE.EQ.1 ) THEN
            WRITE ( 6,70 )
   70       FORMAT ( /1X,'DEBRIS ANALYSIS SPECIFIED BUT RESPONSE FILE',
     1                ' IS FOR METECROIDS ')
          ELSE
```

```
            WRITE ( 6,80 )
   80       FORMAT (/1X,'METEOROID ANALYSIS SPECIFIED BUT RESPONSE',
     1              ' FILE IS FOR DEBRIS' )
         END IF
C
         WRITE ( 6,90 )
   90    FORMAT ( /1X,'DO YOU WISH TO CONTINUE (CR=NO) >',$)
         READ ( 5,30 ) ANSWER
C
         IF ( ANSWER(1:1).EQ.'Y' ) THEN
            GO TO 10
         ELSE
            STOP
         END IF
C
      END IF
C
C  Read in the impact angle information
C
      READ (2) NB,BINC
C
C  Read in the impact velocity information
C
      READ (2) NV,VINC
C
C  Initialize RESPONSE to 0.0
C
      DO 200 I=1,NC
         DO 150 J=1,NB
            DO 100 K=1,NV
               RESPONSE ( K,J,I ) = 0.0
  100       CONTINUE
  150    CONTINUE
  200 CONTINUE
C
C  Read in the critical diameter data
C
C  Loop thru the property id's
C
      DO 400 I=1,NC
C
C  Loop thru the impact angles
C
         DO 300 J=1,NB
C
C  Loop thru the impact velocities
C
            DO 250 K=1,NV
C
C  Store the critical diameter in response
C
               READ (2) RESPONSE(K,J,I)
  250       CONTINUE
  300    CONTINUE
  400 CONTINUE
C
C  Close the file and return
C
      CLOSE ( UNIT=2,STATUS='KEEP' )
C
C  Write Rfile to summary file
C
      WRITE ( 10,500 )RFILE
  500 FORMAT(1X,'RESPONSE OUTPUT FILE = ',A )
C
      RETURN
C
      END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE CRITDIA
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Critdia determines the diameter of the sphere that just penetrates
C  the given wall configuration at the given impact angle and velocity.
C  The subroutine performs a linear interpolation using the appropriate
C  values from the Response array to estimate the critical diameter.
C
C  note: for variables contained in the common block refer to the main
C        listing for definition
C
C
C  Variable list
C
C    beta = impact angle , measured from the normal , deg
C    b1 = the impact angle nearest to the actual impact angle
C         in the Response array , but less than the actual , deg
C    ib1 = location of b1 in the Response array
C    ib2 =    "      " b2  "  "     "       "
C    iv1 =    "      " v1  "  "     "       "
C    iv2 =    "      " v2  "  "     "       "
C    r1 =  intermediate variable
C    r2 =      "           "
C    r11 = value in Response array at location iv1,ib1,pid
C    r12 =  "     "    "        "   "    "     iv1,ib2,pid
C    r21 =  "     "    "        "   "    "     iv2,ib1,pid
C    r22 =  "     "    "        "   "    "     iv2,ib2,pid
C    v1 = impact velocity nearest the actual impact velocity in the
C         Response array , but still less than the actual
C
C
C
C
      INTEGER*2 IB1,IB2,IV1,IV2,PID
C
      INCLUDE 'COMMON2.BLK'
C
      PARAMETER (PI=3.1415926536)
C
C  Determine the location of the nearest velocity to the actual velocity
C  in the Response array, but still less then the actual
C
      IV1=VR/VINC
C
C  Check that the location is inside the array
C
      IF ( IV1.LT.1 .OR. IV1.GT.NV ) THEN
         WRITE ( 6,10 )VR
   10    FORMAT ( /1X,'THE RELATIVE VELOCITY (VR) IS OUTSIDE OF THE',
     1            ' RESPONSE ARRAY BOUNDS VR (KM/SEC) = ',E12.5)
         STOP
      END IF
C
C  Set the location of the velocity just greater than the actual velocity
C
      IV2=IV1+1
C
```

```
C  Check that the location is inside the array
C
      IF ( IV2.GT.NV ) IV2=IV1-1
C
C  Calculate the value of the velocity at location iv1
C
      V1=IV1*VINC
C
C  Determine the impact angle in deg
C
      BETA=ACOS(CBETA)*180.0/PI
C
C  Determine the location of the nearest impact angle to the actual
C  impact angle in the Response array, but still less than the actual
C
      IB1=BETA/BINC+1
C
C  Check that the location is inside the array
C
      IF ( IB1.LT.1 .OR. IB1.GT.NB ) THEN
         WRITE ( 6,20 )BETA
 20      FORMAT ( /1X,'IMPACT ANGLE (BETA) IS OUTSIDE THE BOUNDS OF',
     1              ' THE RESPONSE ARRAY BETA (DEG) = ',E12.5)
         STOP
      END IF
C
C  Set the location of the impact angle in the Response array that is
C  just greater than the actual
C
      IB2=IB1+1
C
C  Check that the location is inside the array
C
      IF ( IB2.GT.NB ) IB2=IB1-1
C
C  Calculate the value of the impact angle at location ib1 in the Response
C  array
C
      B1=(IB1-1)*BINC
C
C  Determine the property id
C
      PID=ID(2,NEL)
C
C  Check that pid is within bounds of the response array
C
      IF ( PID.GT.NC ) THEN
         WRITE ( 6,30 ) PID
 30      FORMAT ( /1X,'NO DATA EXISTS FOR PORPERTY ID ',I4,'IN THE ',
     1              'RESPONSE FILE' )
         STOP
      END IF
C
C  Get the four values that surround the actual value in the Response
C  array
C
      R11=RESPONSE(IV1,IB1,PID)
      R12=RESPONSE(IV1,IB2,PID)
      R21=RESPONSE(IV2,IB1,PID)
      R22=RESPONSE(IV2,IB2,PID)
C
C  Using linear interpolation, estimate the critical diameter
C
```

```
      R1=(R12-R11)*((BETA-B1)/BINC)+R11
      R2=(R22-R21)*((BETA-B1)/BINC)+R21
C
      DIAM=(R2-R1)*((VR-V1)/VINC)+R1
C
C  Finished , return
C
      RETURN
C
      END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE FLUX
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Flux calculates the meteoroid or debris flux for the given critical
C  diameter based on analysis type.
C
C
C  note: for variables contained in the common block referr to the main
C        listing for definition
C
C  Variable List
C
C   ddiam = diam in double precision , cm
C   ge = gravity focusing factor
C   intercept = intercept of the flux equation
C   mass = critical meteoroid mass, g
C   mden = meteoroid density, g/cc
C   re = earth's radius (including 100km atmosphere), km
C   slope = slope of the flux equation
C
C
      INCLUDE 'COMMON2.BLK'
C
      REAL*8 DDIAM,GE,INTERCEPT,LD,MASS,MDEN,PI,RE,SLOPE
C
      PARAMETER (PI=3.141592653589793238D0)
C
C  Set  mden
C
C
      MDEN=0.50D0
C
C  Calculate the focusinng factor, equation
C  is from JSC-30000
C
      RE=6478.0D0
      GE=0.568D0+0.432D0*(RE/(RE+ALT))
C
C  Convert diam to double precision
C
      DDIAM=DIAM
C
C  Calculate the flux
C
      IF ( ITYPE.EQ.1 ) THEN
C
C  For debris use JSC-20001, use stated equations for diameters
C  less then 1 cm , for those greater use third order fit of the
C  curve for region up to 5 cm .
C
C  The log of the flux varies linearly between 400 and 500 km according
C  to D Kesseler of JSC.
C
         LD=DLOG10(DDIAM)
         IF ( DIAM.LT.5.0 ) THEN
            IF ( DIAM.LT.1.0 ) THEN
               SLOPE=-0.0010D0*ALT-2.0200D0
            ELSE
               SLOPE=-0.0022D0*ALT-0.1400D0
            END IF
            INTERCEPT=+0.0036D0*ALT-7.26D0
            FLX=10.0D0**(SLOPE*LD+INTERCEPT)
         ELSE
            WRITE ( 6,100  )
 100        FORMAT ( /1X,'DIAMETER IS GREATER THAN 5 CM LIMIT')
            STOP
         END IF
C
C  Correct Flux for differance in Boeing and Nasa definetion
C
         FLX=FLX*4.0D0
C
      ELSE
C
C  For meteoroids use JSC-3000,  E-06g < mass < 1g
C
         MASS=PI*(DDIAM**3)/6.0D0*MDEN
         FLX=10.0D0**(-14.37D0-1.213D0*DLOG10(MASS))
C
C  Account for earth shielding and gravity focusing , also convert to
C  number of impacts per sq-m per year
C
         FLX=FLX*GE*3.15576D07
C
      END IF
C
      RETURN
C
      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE SUPER (HISTORY)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Super writes out the Supertab universal file that contains the
C  probability of penetration per surface area data.
C
C
C  Variable list
C
C    answer = character string representing user input
C    ids = dataset id , from Supertab manual
C    ilt = leading and trailing -1
C    i61 = field value from Supertab manual
C    i62 =   "     "    "       "      "
C    i63 =   "     "    "       "      "
C    i64 =   "     "    "       "      "
C    i65 =   "     "    "       "      "
C    i66 =   "     "    "       "      "
C    i71 =   "     "    "       "      "
C    i72 =   "     "    "       "      "
C    i73 =   "     "    "       "      "
C    i91 =   "     "    "       "      "
C    i92 =   "     "    "       "      "
C    ofile = output filename
C    ppen = probability of penetration per surface area , %/sq-meter
C    r81 = field value from Supertab manual
C
C
      INCLUDE 'COMMON2.BLK'
C
      CHARACTER*80 ANSWER,OFILE
C
      REAL*8 HISTORY(IEL)
C
C  Read in output filename, set default to out.uni
C
   10 WRITE ( 6,20 )
   20 FORMAT(/1X,'SUPERTAB INPUT FILENAME ? <CR=OUT.UNI>',$)
      READ ( 5,30 ) OFILE
   30 FORMAT( A )
C
      IF ( OFILE(1:1).EQ.' ' ) OFILE='OUT.UNI'
C
C  Open the output file
C
      OPEN ( UNIT=2,FILE=OFILE,STATUS='NEW',IOSTAT=IER,ERR=40 )
C
      GO TO 70
C
C  Error control on open
C
   40 IF ( IER.EQ.2013 ) THEN
         WRITE ( 6,50 )
   50    FORMAT ( /1X,'FILE ALREADY EXISTS OK TO OVERWRITE (CR=YES) >')
         READ ( 5,30 ) ANSWER
C
         IF ( ANSWER(1:1).EQ.'Y' .OR. ANSWER(1:1).EQ.' ' ) THEN
```

```
            OPEN ( UNIT=2,FILE=OFILE,STATUS='UNKNOWN',IOSTAT=IER,
     1             ERR=40 )
            REWIND 2
         END IF
      ELSE
         WRITE ( 6,60 )
   60    FORMAT ( /1X,'UNABLE TO OPEN FILE ' )
         GO TO 10
      END IF
C
C  Write the leading -1
C
   70 ILT=-1
      WRITE ( 2,100 ) ILT
  100 FORMAT (I6)
C
C  Write the dataset id
C
      IDS=56
      WRITE ( 2,110 ) IDS
  110 FORMAT (I6)
C
C  Write the id information
C
      IF ( ITYPE.EQ.1 ) THEN
         WRITE ( 2,120 )
      ELSE
         WRITE ( 2,125 )
      END IF
C
  120 FORMAT ( 1X,'MAN-MADE ORBITAL DEBRIS ANALYSIS')
  125 FORMAT ( 1X,'METEOROID ANALYSIS')
C
      WRITE ( 2,130 )
  130 FORMAT(1X,'PROBABILITY OF PENETRATION (%) PER SQ-METER')
C
      DO 150 I=1,3
         WRITE ( 2,140 )
  140    FORMAT ('NONE')
  150 CONTINUE
C
C  Write out record 6
C
C  Set the 6 field values
C
      I61=1
      I62=0
      I63=1
      I64=1
      I65=2
      I66=1
C
      WRITE ( 2,160 ) I61,I62,I63,I64,I65,I66
  160 FORMAT ( 6I10 )
C
C  Write record 7
C
C  Set the 3 field values
C
      I71=1
      I72=1
      I73=1
C
```

```
      WRITE ( 2,170 ) I71,I72,I73
  170 FORMAT ( 3I10 )
C
C  Write out record 8
C
C  Set the single field value
C
      R81=0.0
C
      WRITE ( 2,180 ) R81
  180 FORMAT (E13.5)
C
C  Write out record 9 & 10 for each element
C
      DO 250 I=1,NELM
C
C  Calculate the probability of penetration per surface area
C
        PPEN=HISTORY(I)*100.0 /AREA(I)
C
C  If the value of ppen is < .000001 no need to write data out
C
        IF ( PPEN.LT.1.0E-06 ) GO TO 250
C
C  Write record 9
C
C  Set the 2 field values
C
        I91=ID(1,I)
        I92=1
C
        WRITE ( 2,190 )I91,I92
  190   FORMAT ( 2I10 )
C
C  Write record 10
C
        WRITE ( 2,200 ) PPEN
  200   FORMAT ( E13.5 )
C
C  Next element
C
  250 CONTINUE
C
C  Write out trailing -1
C
      WRITE ( 2,260 ) ILT
  260 FORMAT ( I6 )
C
C  Close the file
C
      CLOSE ( UNIT=2 , STATUS='KEEP' )
C
C  Finished return
C
      RETURN
C
      END
```

COMMON2.BLK

```
C
C   Common Block for Bumper Ver 4.0 5/25/87
C
C   ielm - max number of elements
C   ith - max number of threats
C
      PARAMETER (IELM=9000,ITH=400)
C
      INTEGER*2 IT,ITYPE,NB,NC,NEL,NELM,NT,NV,EXPOSED(ITH),
     1          POINT(IELM,ITH)
C
      INTEGER*4 NR,ID(2,IELM),RANGE(2,50)
C
      REAL*4 BINC,CBETA,DIAM,VINC,AREA(IELM),GEOMETRY(IELM,ITH),
     1       RESPONSE(70,90,5),THREAT(4,ITH)

      REAL*8 ALT,ETIME,FLX
C
      COMMON ALT,BINC,CBETA,DIAM,ETIME,FLX,IT,ITYPE,NB,NC,NEL,NELM,
     1       NR,NT,NV,VR,VINC,AREA,EXPOSED,GEOMETRY,ID,POINT,RANGE,
     2       RESPONSE,THREA
```

APPENDIX E

Source code for CONTOUR Module

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                   C
C      CONTOUR VER 2.0  6/5/87      C
C                                   C
C       BOEING AEROSPACE CO.        C
C                                   C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  CONTOUR develops data for producing design contour plots for
C  use in designing spacecraft subject to meteoroid debris impact.
C  It is a modification to the BUMPER code. THe modification involved
C  making the RESPONSE code a subroutine and expanding the user input.
C  The code ddevelops the data for a specific range of element ids
C  and a single property id.
C
C  The code was developed under NASA contract ' Integrated Wall Design
C  Guide and Penetration Control Plan', by M.A.Wright and A.R.Coronado.
C
C
C
C
C  Common Block Variable list
C
C  Scalers
C
C    alt = operating altitude , km
C    binc = impact angle (beta) increment , deg
C    cbeta = cosine of beta
C    diam = critical diameter , cm
C    etime = spacecraft exposure time , years
C    flx = number of impacts per projected area per year of diameter D
C           or larger
C    itype = analysis type , 1- debris, 2-meteoroids
C    nb = number of angles in the response array
C    nel = current element number
C    nelm = total number of elements
C    nr = number of element ranges to sum over
C    nt = number of threat cases
C    nv = number of velocities in the response array
C    vr = impact (relative) velocity , km/sec
C    vinc = impact (relative) velocity increment , km/sec
C
C  Arrays
C
C    area = array containing the value of the surface area for each
C           element, sq-meters
C    exposed = list of the number of exposed elements for each threat
C              angle.
C    geometry = array containing the values of the cosine of the impact
C               angle for each exposed element for each threat angle.
C    id = array containing the values of the element and property id
C         for each element
C         1- id
C         2- pid
C    point = array of the element numbers corresponding to the elements
C            in the geometry array.
C    range = array containing the starting and ending elment id for
C            each range to sum over
C            1-starting id
C            2- ending id
C    rtable = array containing the values of the critical diameter as
```

```
C                 a function of impact angle and velocity.
C                 (vr,beta,pid)
C      threat = array containg threat information
C               1-theta angle, rad
C               2-phi angle, rad
C               3-vr, km/sec
C               4-prob
C      history = list containing the  running sum of the fa term for each
C                element
C
C
C    Main Program Variable List
C
C    Scalers
C
C      answer = user input
C      ctype = configuration type
C              1- single plate
C              2- double plate
C      fa = product of the threat probability, flux and cosine(beta),
C           impacts/yr/sq-meter
C      kc = counter
C      mli = logical variable used to determine if multi-layer
C            insulation is to be included
C      nst = number of shield thickness's
C      nwt = number of vessel wall thickness's
C      pfunc = penetration function
C              1- original
C              2- pen4
C              3- regression
C      pid = property id
C      pnp = probabibltiy of no penetration, %
C      prob = probabiltiy of threat case i occuring
C      rat = ratio of the shiled thickness to total thickness
C      shthk = shield thickness , in
C      stand = shield stand-off distance, in
C      sum = sum of the history array for the specific range
C      t1inc = shield thickness increment , in
C      t1max = maximum shield thickness , in
C      t1min = minimum shield thickness , in
C      t2inc = shield thickness increment , in
C      t2max = maximum shield thickness , in
C      t2min = minimum shield thickness , in
C      vwthk = vessel wall thickness, in
C
C
C    Arrays
C
C      bhard = array containing the brinnel hardnes values for the shield
C              vessell wall and projectile
C      C = arrary containing the speed of sound values for the shield
C          vessel wall and projectile, ft/sec
C      dens = array containing the densisty of the shield vessel wall
C             and projectile, lbs/in**3
C      fsu = array containing the allowable shear stress for the shield
C            vessel wall and projectile, psi
C      ftu = array containing the allowable tensile ultimate stress for
C            the shield vessel wall and projectile, psi
C      fy = array containing the allowable yield stress for the shield
C           vessel wall and projectile, psi
C      wilkc = array containg Wilikinson's constant for the shield vessel
C              wall and projectile
C
```

```
C
      INCLUDE 'COMMON3.BLK'
C
      CHARACTER*80 ANSWER
C
      INTEGER*2 CTYPE,PID,PFUNC
C
      LOGICAL MLI
C
      REAL*4 STAND,T1INC,T1MAX,T1MIN,T2INC,T2MAX,T2MIN,BHARD(3),C(3),
     1       DENS(3),FSU(3),FTU(3),FY(3),SHPV(3),WILKC(3)
C
      REAL*8 FA,PROB,PNP,SUM,HISTORY(IEL)
C
C Write header to screen,
C
      CALL HEADER ( ITYPE )
C
C Read in screen inputs
C
      CALL INPUT ( CTYPE,MLI,PID,PFUNC,STAND,T1INC,T1MAX,T1MIN,T2INC,
     1             T2MAX,T2MIN,BHARD,C,DENS,FSU,FTU,FY,SHPV,WILKC )
C
C Read in the GEOMETRY output file
C
      CALL GEOREAD
C
C Set the number of shield and vessel wall thickness cases
C
      NST = NINT((T1MAX-T1MIN)/T1INC)+1
      NWT = NINT((T2MAX-T2MIN)/T2INC)+1
C
C Initialize counter to 1
C
      KC = 1
C
C Loop through the various wall configurations
C
      DO 1000 I=1,NWT
C
C Set vessel wall thickness
C
         VWTHK=T2MIN+(I-1)*T2INC
C
         DO 900 J=1,NST
C
C Set the shield thickness
C
            SHTHK=T1MIN+(J-1)*T1INC
C
C Check that configuration is within the limits
C
            IF ( CTYPE.NE.1 ) THEN
               RAT=SHTHK/(SHTHK+VWTHK)
               IF ( RAT.GT.0.50 .OR. RAT.LT.0.10 ) GO TO 900
            END IF
C
C Build the response array
C
            CALL RESPONSE (BINC,CTYPE,ITYPE,MLI,NB,NV,PID,PFUNC,SHTHK,
     1                     STAND,VWTHK,VINC,BHARD,C,DENS,FSU,FTU,FY,
     2                     RTABLE,SHPV,WILKC )
C
```

```
C  Initialize History to 0.0
C
            DO 50 K=1,NELM
               HISTORY(K)=0.0D0
 50         CONTINUE
C
C  Initialize Sum  to 0.0
C
            SUM =0.0D0
C
C  Determine the penetrating flux for each element, for each threat
C  angle
C
            DO 200 K=1,NT
C
C  Set the threat index and get the impact velocity and the threat
C  probability from the threat array
C
               VR=THREAT(3,K)
               PROB=THREAT(4,K)
C
C  Evaluate each exposed element
C
               DO 100 L=1,EXPOSED(K)
C
C  Set the element number
C
                  NEL=POINT(L,K)
C
                  IF ( RANGE(1).GT.ID(1,NEL) ) GO TO 100
                  IF ( RANGE(2).LT.ID(1,NEL) ) GO TO 100
C
C  Check that proprety id specified matches elements property id
C  if not skip over element.
C
                  IF ( PID.NE.ID(2,NEL) ) THEN
                     GO TO 100
                  END IF
C
C  Get the cosine of the impact angle from the Geometry array.
C
                  CBETA=GEOMETRY(L,K)
C
C  Determine the diameter of the sphere that just penetrates the wall
C
                  CALL CRITDIA
C
C  Calculate the flux of the critical diameter using the appropriate
C  flux equation based on the analysis type.
C
                  CALL FLUX
C
C  Multiply the flux by the probability of the threat angle and the
C  cosine of the impact angle, this determines the number of
C  penetrations per year per element surface area for the current
C  element and the current threat angle.
C
                  FA=FLX*PROB*CBETA
C
C  Store the running sum of the FA term for each element in the History
C  array. This value represents the total number of penetrations for
C  a given element per year per element surface area.
C
```

```
                  HISTORY(NEL)=HISTORY(NEL)+FA
C
C  Next element
C
  100          CONTINUE
C
C  Next threat
C
  200       CONTINUE
C
C  Multiply each term in the HISTORY array by the appropriate element
C  surface area.
C
            DO 250 K=1,NELM
               HISTORY(K)=HISTORY(K)*AREA(K)
  250       CONTINUE
C
C  Sum up the HISTORY array by range
C
            DO 310 K=1,NELM
               IF ( RANGE(1).GT.ID(1,K) ) GO TO 310
               IF ( RANGE(2).LT.ID(1,K) ) GO TO 310
               IF ( PID.NE.ID(2,K) ) GO TO 310
               SUM=SUM+HISTORY(K)
  310       CONTINUE
C
C  Calculate the probabilty of no penetration for each range using
C  a Possion model
C
            PNP=(DEXP(-SUM*ETIME))*100.D0
C
C  Write the probability value to the screen and the summary file.
C
            WRITE (  6,390 ) KC,SHTHK,STAND,VWTHK,PNP
            WRITE ( 10,390 ) KC,SHTHK,STAND,VWTHK,PNP
  390       FORMAT ( /1X,I4,3F7.4,F12.5 )
C
            KC=KC+1
C
C  Next shield thickness
C
  900    CONTINUE
C
C  Next vessel wall thickness
C
 1000 CONTINUE
C
C  Close summary file
C
       CLOSE ( UNIT=10,STATUS='KEEP' )
C
C  Finished
C
      END
```

```
C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
      SUBROUTINE HEADER ( ITYPE )
C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C
C  Header writes out the program header and reads in the summary file
C  name and analysis type ( meteoroid or debris ).
C
C  Variable list
C
C    answer = user input
C
      CHARACTER*80 ANSWER
C
C  Write Header to screen
C
      WRITE ( 6,10 )
   10 FORMAT ( /1X,'##############################',//,5X,'CONTOUR ',
     1           'Version 2.0',//,5x,'last update 6/5/87',//1x,
     2           '##############################' )
C
C  Read in summary output filename, set default to contour.sum
C
   20 WRITE ( 6,30 )
   30 FORMAT ( /1X,'SUMMARY OUTPUT FILENAME (CR=CONTOUR.SUM) : ',$)
      READ ( 5,'(A)' )ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' ) ANSWER='CONTOUR.SUM'
C
C  Open sfile
C
      OPEN ( UNIT=10,FILE=ANSWER,STATUS='UNKNOWN',ERR=40 )
C
      REWIND 10
C
      GO TO 60
C
C  Error control
C
   40 WRITE ( 6,50 )
   50 FORMAT (/1X,'UNABLE TO OPEN FILE ' )
      GO TO  20
C
C  Determine analysis type, set default to 1 (debris)
C
   60 WRITE ( 6,70 )
   70 FORMAT (/1X,'ANALYSIS TYPE ?',/,2X,'1-DEBRIS <CR> ',/,2X,
     1          '2-METEOROIDS',/,1X,'ANSWER 1 OR 2 : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         ITYPE=1
      ELSE
         READ ( ANSWER(1:4),'(BN,I4)',ERR=60 )ITYPE
      END IF
C
C  Check that input was correct
C
      IF ( ITYPE.EQ.1 .OR. ITYPE.EQ.2 ) THEN
         CONTINUE
      ELSE
         WRITE ( 6,80 )
   80    FORMAT ( /1X,'INCORRECT INPUT' )
         GO TO 60
      END IF
C
C  Write analysis type and etime to summary file
C
      WRITE ( 10,10 )
C
      IF ( ITYPE.EQ.1 ) THEN
         WRITE ( 10,90 )
   90    FORMAT ( /1X,'MAN-MADE ORBITAL DEBRIS ANALYSIS')
      ELSE
         WRITE ( 10,100 )
  100    FORMAT ( 1X,'METEOROID ANALYSIS' )
      END IF
C
C  Finished
C
      RETURN
C
      END
```

```
C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
      SUBROUTINE INPUT ( CTYPE,MLI,PID,PFUNC,STAND,T1INC,T1MAX,T1MIN,
     1                   T2INC,T2MAX,T2MIN,BHARD,C,DENS,FSU,FTU,FY,
     2                   SHPV,WILKC )
C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C  Input reads in the user defined inputs and writes them out to the
C  summary file. It is a modified version of the combined Resposne and
C  Bumper codes input subroutines.
C
C  For variables in the common block see main listing
C
C  Variable list
C
C    ncases = potential numeber of wall configurations
C    metric = logical variable used to determine if input units are in
C             the metric system
C    lc = number of materials red in from the MAT.PRP file
C    answer = user input
C    luniits = lenght units in or cm
C
C  Array list
C
C    dns = temporary array containing the density values in the MAT.PRP
C          file, lbs/in**3
C    con = temporary array containing the Wilkinson's constant in the
C          MAT.PRP file, km/sec
C    hrd = temporary array containing  Brinnel Hardness values in the
C          MAT.PRP file, unitless
C    shr = temporary array containing the Shear stress allowable values
C          in the MAT.PRP file , psi
C    shk = temporary array containg the shock projectile velocities in
C          in the MAT.PRP file
C    ssnd = temporary array containing the speed of sound values in the
C           MAT.PRP file, ft/sec
C    ult = tempoary array containing the ultimate tensile stress
C          allowables in the MAT.PRP file
C    yld = temporary array containing  yield stress allowables in the
C          MAT.PRP file
C
C
      INCLUDE 'COMMON3.BLK'
C
C
      INTEGER*2 CTYPE,PID,PFUNC
C
      LOGICAL MLI
C
      REAL*4 STAND,T1INC,T1MAX,T1MIN,T2INC,T2MAX,T2MIN,BHARD(3),C(3),
     1       DENS(3),FSU(3),FTU(3),FY(3),SHPV(3),WILKC(3)
C
      CHARACTER*2 LUNITS
      CHARACTER*12 MNAM(10)
      CHARACTER*20 ANSWER
C
      INTEGER*2 LC,MAT(3)
C
      LOGICAL METRIC
C
      REAL*4 DNS(10),CON(10),HRD(10),SHR(10),SHK(10),SSND(10),ULT(10),
     1       YLD(10)
C
C Initialize variables
C
      LC=1
      MLI=.TRUE.
      METRIC=.TRUE.
C
C Determine the spacecraft exposure time , set default to 10 years
C and write out to summary file
C
   10 WRITE ( 6,20 )
   20 FORMAT (/1X,'SPACECRAFT EXPOSURE TIME (YEARS) <CR=10.0> : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         ETIME=10.0D0
      ELSE
         READ ( ANSWER(1:20),'(BN,D20.0)',ERR=10 ) ETIME
      END IF
C
      WRITE ( 10,30 )ETIME
   30 FORMAT ( /1X,'SPACECRAFT EXPOSURE TIME (YEARS) =',F7.2 )
C
C Read in operating altitude , set default to 500 km, and write out to
C summary file
C
   40 WRITE ( 6,50 )
   50 FORMAT ( /1X,'OPERATING ALTITUDE ( 400-500 km )  <CR=500> : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         ALT=500.0D0
      ELSE
         READ ( ANSWER(1:20),'(BN,D20.0)',ERR=40 ) ALT
      END IF
C
C Check that altitude is within range
C
      IF ( ALT.LT.350.0 .OR. ALT.GT.550.0 ) THEN
         WRITE ( 6,60 )
   60    FORMAT ( 1X,'---ERROR--- Altitude outside of range' )
         GO TO 40
      END IF
C
      WRITE ( 10,70 ) ALT
   70 FORMAT ( /1X,'OPERATING ALTITUDE (km) = ',F7.2 )
C
C Read in element range to sum over
C
      WRITE ( 6,80 )
   80 FORMAT ( /1X,'THE PROBABILITY OF NO PENETRATION WILL BE ',
     1         'CALCULATED FOR A SPECIFIC RANGE',/1X,'OF ELEMENT IDS ',
     2         'INPUT THE STARTING AND ENDING ELEMENT ID FOR EACH ',
     3         'RANGE')
C
   90 WRITE ( 6,100 )
  100 FORMAT ( /1X,'STARTING ELEMENT ID : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      READ ( ANSWER(1:20),'(BN,I8)',ERR=90 ) RANGE(1)
C
  110 WRITE ( 6,120 )
  120 FORMAT ( 1X,'ENDING ELEMENT ID   : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      READ ( ANSWER(1:20),'(BN,I8)',ERR=110 ) RANGE(2)
C
C Check that ending id > starting id
C
      IF ( RANGE(1).GT.RANGE(2) ) THEN
         WRITE ( 6,130 )
  130    FORMAT ( 1X,'---ERROR--- Staring ID greater then Ending ID')
         GO TO 90
      END IF
C
C  Write range out to summary file
C
      WRITE ( 10,140 ) RANGE(1),RANGE(2)
  140 FORMAT ( /1X,'STARTING ELEMENT ID =',I8,/1X,'ENDING ELEMENT ID ',
     1         '  =',I8 )
C
C  Read in the  material properties file.
C
      OPEN (UNIT=2,FILE='MAT.PRP',STATUS='OLD')
C
  150 READ ( 2,'(A,BN,8E12.5)',END=160 )MNAM(LC),DNS(LC),YLD(LC),
     1                                  ULT(LC),SHR(LC),CON(LC),
     2                                  SSND(LC),SHK(LC),HRD(LC)
C
      LC=LC+1
      GO TO 150
C
  160 CLOSE ( UNIT=2,STATUS='KEEP' )
      LC=LC-1
C
C  Determine the type of units for input.
C
      WRITE ( 6,170 )
  170 FORMAT (/1X,'INPUT IN METRIC OR ENGLISH UNITS  <CR>=METRIC : ',$)
      READ ( 5,'(A)' )ANSWER
C
      IF ( ANSWER(1:1).EQ.'E' ) METRIC=.FALSE.
C
C  Set the units
C
  180 IF ( METRIC ) THEN
          LUNITS='CM'
      ELSE
          LUNITS='IN'
      END IF
C
C  Read in property id
C
  185 WRITE ( 6,190 )
  190 FORMAT ( /1X,'PROPERTY ID = <CR=1> :',$)
      READ ( 5,'(A)' ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         PID=1
      ELSE
         READ ( ANSWER(1:20),'(BN,I4)',ERR=185 ) PID
      END IF
C
      IF ( PID.GT.10 .OR. PID.LT.1 ) THEN
         WRITE ( 6,'(/1X,''INCORRECT INPUT'' )' )
         GO TO 185
      END IF
C
      WRITE ( 10,'(/1X,''PROPERTY ID = '',I4 )' ) PID
C
C  Determine configuration type and write out to summary file
C
  200 WRITE ( 6,210 )
  210 FORMAT ( /1X,'CONFIGURATION TYPE',/,5X,'1- SINGLE PLATE ',/,
     1         5X,'2- DOUBLE PLATE <CR>',/,1X,'ANSWER (1 or 2) : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         CTYPE=2
      ELSE
         READ ( ANSWER(1:4),'(BN,I4)',ERR=200 ) CTYPE
      END IF
C
C  Check that the input was correct
C
      IF ( CTYPE.LT.1 .OR. CTYPE.GT.2 ) THEN
         WRITE ( 6,220 )
  220    FORMAT( /1X,'INCORRECT INPUT' )
         GO TO 200
      END IF
C
      IF ( CTYPE.EQ.1 ) THEN
         WRITE ( 10,230 )
  230    FORMAT ( /1X,'SINGLE PLATE ' )
      ELSE
         WRITE ( 10,240 )
  240    FORMAT ( /1X,'DOUBLE PLATE ANALYSIS' )
      END IF
C
C  For single plate configuration skip down to the vessel wall material
C
      IF ( CTYPE.EQ.1 ) GO TO 430
C
C  Determine which double wall penetration function to use and write
C  out to summary file
C
  250 WRITE ( 6,260 )
  260 FORMAT ( /1X,'PENETRATION FUNCTION ',/,5X,'1-ORIGINAL <CR>',/,
     1         5X,'2-PEN4',/,5X,'3-REGRESSION',/,1X,'ANSWER (1-3) : ',
     2         $)
C
      READ ( 5,'(A)' ) ANSWER
      IF ( ANSWER(1:1).EQ.' ' ) THEN
         PFUNC=1
      ELSE
         READ ( ANSWER(1:20),'(BN,I4)',ERR=250 ) PFUNC
      END IF
C
C  Check Input
C
      IF ( PFUNC.LT.1 .OR. PFUNC.GT.3 ) GO TO 250
C
      IF ( PFUNC.EQ.1 ) THEN
         WRITE ( 10,270 )
  270    FORMAT ( /1X,'ORGINAL PENETRATION FUNCTION' )
      ELSE IF ( PFUNC.EQ.2 ) THEN
         WRITE ( 10,280 )
  280    FORMAT ( /1X,'PEN4 PENETRATION FUNCTION' )
```

```
      ELSE IF ( PFUNC.EQ.3 ) THEN
         WRITE ( 10,290 )
  290    FORMAT ( /1X,'REGRESSION PENETRATION FUNCTION' )
      END IF
C
C  Determine the shield material.
C
  300 WRITE ( 6,310 )
  310 FORMAT (/1X,'SHIELD MATERIAL ')
C
C  Write out the material list.
C
      DO 320 I=1,LC
         WRITE ( 6,315 )I,MNAM(I)
  315    FORMAT ( 3X,I2,'- ',A )
  320 CONTINUE
C
C  Set the material default number equal to one.
C
  330 WRITE ( 6,340 )
  340 FORMAT (1X,'SELECT MATERIAL NUMBER <CR>=1 : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      IF ( ANSWER(1:1).EQ.' ') THEN
         MAT(1)=1
      ELSE
         READ ( ANSWER(1:4),'(BN,I4)',ERR=330 ) MAT(1)
      END IF
C
C  Check that the value read in is  contained in the list.
C
      IF ( MAT(1) .LT.1 .OR. MAT(1).GT. LC ) GO TO 330
C
C  Write shield material out to summary file
C
      WRITE ( 10,350 ) MNAM(MAT(1))
  350 FORMAT ( /1X,'SHILED MATERIAL = ',A )
C
C  Determine the shield minimum, maximum, and increment thickness.
C
  360 WRITE ( 6,370 ) LUNITS
  370 FORMAT ( /1X,'MINIMUM SHIELD THICKNESS (',A,') = : ',$)
      READ ( 5,*,ERR=360 ) T1MIN
C
  380 WRITE ( 6,390 ) LUNITS
  390 FORMAT ( /1X,'MAXIMUM SHIELD THICKNESS (',A,') = : ',$)
      READ ( 5,*,ERR=380 ) T1MAX
C
  400 WRITE ( 6,410 ) LUNITS
  410 FORMAT ( /1X,'INCREMENT SHIELD THICKNESS (',A,') = : ',$)
      READ ( 5,*,ERR=400 ) T1INC
C
C  Write out values to summary file
C
      WRITE ( 10,420 ) LUNITS,T1MIN,T1MAX,T1INC
  420 FORMAT ( /1X,'MINIMUM MAXIMUM AND INCREMENT SHIELD THICKNESS (',
     1         A,') = ',/1X,3F7.4 )
C
C  Determine the vessel wall material.
C
  430 WRITE ( 6,440 )
  440 FORMAT (/1X,'VESSEL WALL MATERIAL ' )
C
```

```
      DO 450 I=1,LC
       WRITE ( 6,315 ) I,MNAM(I)
  450 CONTINUE
C
  460 WRITE ( 6 ,340 )
      READ ( 5,'(A)'  ) ANSWER
C
      IF ( ANSWER(1:1) .EQ. ' ' ) THEN
         MAT(2)=1
      ELSE
         READ ( ANSWER (1:4),'(BN,I4)',ERR=460 ) MAT(2)
      END IF
C
      IF ( MAT(2).LT.1 .OR. MAT(2).GT.LC ) GO TO 460
C
C  Write vessel wall material out to summary file
C
      WRITE ( 10,470 ) MNAM(MAT(2))
  470 FORMAT ( /1X,'VESSEL WALL MATERIAL = ',A )
C
C  Determine the minimum , maximum, and increment vessel wall thickness
C
  480 WRITE ( 6,490 )LUNITS
  490 FORMAT (/1X,'MINIMUM VESSEL WALL THICKNESS (',A,') = : ',$)
      READ ( 5,*,ERR=480 ) T2MIN
C
  500 WRITE ( 6,510 ) LUNITS
  510 FORMAT (/1X,'MAXIMUM VESSEL WALL THICKNESS (',A,') = : ',$)
      READ ( 5,*,ERR=500 ) T2MAX
C
  520 WRITE ( 6,530 ) LUNITS
  530 FORMAT (/1X,'INCREMENT VESSEL WALL THICKNESS (',A,') = : ',$)
      READ ( 5,*,ERR=520 ) T2INC
C
C  Check if number of potential cases is ok
C
      IF ( T1INC.EQ.0.0 ) T1INC=1.0
      NCASES=((T2MAX-T2MIN)/T2INC+1)*((T1MAX-T1MIN)/T1INC+1)
C
      WRITE ( 6,540 ) NCASES
  540 FORMAT (/1X,'THE NUMBER OF POTENTIAL CASES IS ',I6,/1X,'DO YOU ',
     1          'WISH TC CONTINUE ? <CR=YES>  : ',$)
      READ ( 5,'(A)' ) ANSWER
C
      IF ( ANSWER(1:1).EQ.'N' )  GO TO 300
C
C  Write values out to the summary file
C
      WRITE ( 10,550 ) LUNITS,T2MIN,T2MAX,T2INC
  550 FORMAT ( /1X,'MINIMUM MAXIMUM AND INCREMENT VESSEL WALL ',
     1          'THICKNESS (',A,') = ',/1X,3F7.4 )
C
C  For single plate configuration skip stand-off and mli
C
      IF ( CTYPE.EQ.1 ) GO TO 610
C
C  Determine the shield stand-off distance and write out to the summary
C  file.
C
  560 WRITE ( 6,570 )LUNITS
  570 FORMAT ( /1X,'SHIELD STAND-OFF (',A,') = : ',$)
      READ ( 5,*,ERR=560 ) STAND
C
```

```
      WRITE ( 10,580 ) LUNITS,STAND
  580 FORMAT ( /1X,'SHIELD STAND-OFF (',A,') = ',F12.5 )
C
C  Determine if MLI is to be included, but not for the pen4 penetration
C  function
C
      IF ( PFUNC.EQ.2 ) GO TO 610
C
      WRITE ( 6,590 )
  590 FORMAT (/1X,'INCLUDE 30 LAYERS OF MLI BETWEEN PLATES <CR>=Y : ',
     1       $)
      READ ( 5,'(A)' )ANSWER
      IF ( ANSWER(1:1).EQ.' ' .OR. ANSWER(1:1).EQ.'Y' )MLI=.TRUE.
C
      IF ( MLI ) THEN
         WRITE ( 10,600 )
  600    FORMAT ( /1X,'30 LAYERS OF MULTI-LAYER INSULATION INCLUDED' )
      END IF
C
C  Set the projectile material property based on analysis type
C  For debris  100-0 al, for meteoroids only density is important use
C  use 0.50 g/cc .
C
  610 IF ( ITYPE.EQ.1 ) THEN
         BHARD(3)=23.0
         C(3)=16550.0
         DENS(3)=0.098
         FSU(3)=5000.0
         FTU(3)=13000.0
         FY(3)=9000.0
         SHPV(3)=1.345
         WILKC(3)=.126
      ELSE
         BHARD(3)=0.0
         C(3)=0.0
         DENS(3)=0.50/27.705
         FSU(3)=0.0
         FTU(3)=0.0
         FY(3)=0.0
         SHPV(3)=0.0
         WILKC(3)=0.0
      END IF

C
C  Build the material properties arrays.
C
      DO 650 I=1,2
         BHARD(I)=HRD(MAT(I))
         C(I)=SSND(MAT(I))
         DENS(I)=DNS(MAT(I))
         FSU(I)=SHR(MAT(I))
         FTU(I)=ULT(MAT(I))
         FY(I)=YLD(MAT(I))
         SHPV(I)=SHK(MAT(I))
         WILKC(I)=CON(MAT(I))
  650 CONTINUE
C
C  If the variables were read in in metric units, convert to english.
C
      IF ( METRIC ) THEN
         T1MIN=T1MIN/2.54
         T1MAX=T1MAX/2.54
         T1INC=T1INC/2.54
         T2MIN=T2MIN/2.54
         T2MAX=T2MAX/2.54
         T2INC=T2INC/2.54
         STAND=STAND/2.54
      END IF
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE GEOREAD
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Georead reads in the output file from the GEOMETRY code. This file
C  contains the global threat and element data as well as the list of
C  exposed elements and their impact angles for each threat case.
C
C
C
C  note: for variables contained in the common block refer to main
C        listing for definition
C
C
C  Variable List
C
C    answer = character string representing user input
C    gfile = geometry output filename
C    itf = analysis type contained in the
C
C
      INCLUDE 'COMMON3.BLK'
C
      CHARACTER*80 ANSWER,GFILE
C
      INTEGER*2 ITF
C
C  Read in the GEOMETRY output filename, set the default to geom.dat
C
   10 WRITE ( 6,20 )
   20 FORMAT(/1X,'GEOMETRY OUTPUT FILENAME ? <CR=STATION.GEM> >',$)
      READ ( 5,30 ) GFILE
   30 FORMAT (A)
C
      IF ( GFILE(1:1).EQ.' ' ) GFILE='STATION.GEM'
C
C  Open the file
C
      OPEN (UNIT=2,FILE=GFILE,STATUS='OLD',FORM='UNFORMATTED',ERR=40 )
C
      GO TO  60
C
C  Error control
C
   40 WRITE ( 6,50 )
   50 FORMAT ( /1X,'UNABLE TO OPEN FILE ' )
      GO TO 10
C
C  Read in the analysis type,the number of threat cases, and the
C  number of elements
C
   60 READ (2) ITF,NT,NELM
C
C  Check that the analysis type in the file matches the type input
C  by user.
C
      IF ( ITF.NE.ITYPE ) THEN
         IF ( ITYPE.EQ.1 ) THEN
            WRITE ( 6,70 )
   70       FORMAT ( /1X,'DEBRIS ANALYSIS SPECIFIED BUT FILE IS FOR ',
     1               'METEOROIDS')
         ELSE
            WRITE ( 6,80 )
   80       FORMAT ( /1X,'METEOROID ANALYSIS SPECIFIED BUT FILE IS ',
     1               'FOR DEBRIS' )
         END IF
C
         WRITE ( 6,90 )
   90    FORMAT ( /1X,'DO YOU WISH TO CONTINUE (CR=NO) >',$)
         READ ( 5,30 ) ANSWER
C
         IF ( ANSWER(1:1).EQ.'Y' ) THEN
            GO TO 10
         ELSE
            STOP
         END IF
C
      END IF
C
C   Check that the number of threats and the number of elements are less
C   then the maximum allowed
C
      IF ( NT.GT.ITH ) THEN
         WRITE ( 6,100 )
  100    FORMAT ( /1X,'NUMBER OF THREATS IS GREATER THEN MAX ALLOWED')
         WRITE ( 6,105 )
  105    FORMAT ( 1X,'ARRAY SIZE MUST BE INCREASED & CODE RECOMPILED')
         STOP
      END IF
C
      IF ( NELM.GT.IEL ) THEN
         WRITE ( 6,110 )
  110    FORMAT ( /1X,'NUMBER OF ELEMENTS IS GREATER THEN MAX ALLOWED')
         WRITE ( 6,105 )
         STOP
      END IF
C
C  Initialize the arrays to 0.0
C
      DO 150 I=1,NT
         THREAT(3,I)=0.0
         THREAT(4,I)=0.0
         EXPOSED(I)=0
         DO 140 J=1,NELM
            GEOMETRY(J,I)=0.0
            ID(1,J)=0
            ID(2,J)=0
            POINT(J,I)=0
  140    CONTINUE
  150 CONTINUE
C
C  Read in the Threat data
C
      DO 175 I=1,NT
         READ (2) (THREAT(J,I),J=1,4)
  175 CONTINUE
C
C  Read in the element id, and  property id storing them in the ID
C  array.
C
      DO 200 I=1,NELM
         READ (2) (ID(J,I),J=1,2)
  200 CONTINUE
C
C  Read in the element's surface area storing it in the AREA array.
C
      DO 250 I=1,NELM
         READ (2) AREA(I)
  250 CONTINUE
C
C  Read in the geometry data for the exposed elements
C
      DO 500 I=1,NT
C
C  Read in the threat case and the number of exposed elements
C
         READ (2) IT,EXPOSED(I)
C
C  Loop thru the exposed elements
C
         DO 400 J=1,EXPOSED(I)
C
C  Read in the element number (storing in the POINT array), and the
C  cosine of the impact angle (storing in the GEOMETRY array).
C
            READ (2) POINT(J,I),GEOMETRY(J,I)
C
  400    CONTINUE
C
  500 CONTINUE
C
C  Write gfile to summary file
C
      WRITE ( 10,600 )GFILE
  600 FORMAT ( 1X,'GEOMETRY OUTPUT FILE = ',A )
C
C  Close file
C
      CLOSE ( UNIT=2,STATUS='KEEP' )
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE RESPONSE (AINCR,CTYPE,ITYPE,MLI,NANG,NVEL,PID,PFUNC,
     1                     SHTHK,STAND,VWTHK,VINCR,BHARD,C,DENS,FSU,
     2                     FTU,FY,RTABLE,SHPV,WILKC )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Variable list
C
C    aincr = impact angle increment,deg
C    amax = maximum impact angle,deg
C    amin = minimum impact angle,deg
C    ang = impact angle,deg
C    angr = impact angle,radians
C    answer = character string representing user input
C    ctype = configuration type
C                                1- single plate
C                                2 - double plate
C    dia = projectile diameter,in
C    diam =   "          "    ,cm
C    ic = case counter
C    initial = logical variable used to detrmine if current call to
C              diameter is the initial one for the current angle
C    itype = analysis type,1=space debris
C                             2=meteoroids
C    mli = logical variable used to detrmine if 30 layers of mli is
C          included
C    nang = number of angles to be considered
C    nvel = number of velocities to be considerd
C    pfunc = penetration function
C             1-orginal
C             2-pen4
C             3-regression
C    shthk = shield thickness,in
C    stand = shield stand-off,in
C    vele = impact velocity,ft/sec
C    velm =   "         "     ,km/sec
C    vincr = velocity increment,km/sec
C    vmax = maximum velocity,km/sec
C    vmin = minimum velocity,km/sec
C
C
C  Array list
C
C   bhard = array containing the Brinnel hardness values for the
C           current configuration
C   c = array containing the speed of sound values for the current
C       configuration, ft/sec
C   dens = array containing the density values for the current
C          configuration,lbs/in**3
C   fsu = array containing the shear allowable stress values for the
C         current configuration,psi
C   ftu = array containing the ultimate tensile stress values for the
C         current configuration,psi
C   fy = array containing the yield stress values for the current
C        configuration
C   rtable = array containing the critcal diameters for each case,angle
C            and velocity
C   shpv =array containing the shock projectile velocities for the
C         current configuration
```

```
C     wilkc = array containing the values for Wilkinson's constant for
C              the current configuration,km/sec
C
C
C
C
      CHARACTER*20 ANSWER
C
      DIMENSION RTABLE(70,50,10)
C
      DIMENSION BHARD(3),C(3),DENS(3),FSU(3),FTU(3),FY(3),SHPV(3),
     1          WILKC(3)
C
      INTEGER*2 CTYPE,IC,ITYPE,NANG,NVEL,PID,PFUNC
C
      LOGICAL INITIAL,METRIC,MLI
C
C  Set the angle and velocity limits and increments
C
C  Amin must always =0
C
      AMIN=0.0
C
      AMAX=90.0
      AINCR=5.0
C
C  Determine the number of velocity and angle iterations
C
      NANG = (AMAX-AMIN)/AINCR + 1
C
C  Set velocity values based on analysis type
C
C  Vmin must always = 0
C
      VMIN=0.0
C
      IF ( ITYPE.EQ.1 ) THEN
         VMAX=17.0
         VINCR=0.25
      ELSE
         VMAX=70.0
         VINCR=1.0
      END IF
C
         NVEL=(VMAX-VMIN)/VINCR
C  For the current configuration,determine the critical diameter
C  for each impact angle and velocity
C
      DO 200 I=1,NANG
C
C  Set the angle,in deg and radians
C
         ANG = 0.0 + (I-1)*AINCR
C
C  For angles > 60 deg,set ang=60
C
         IF ( ANG .GT. 60.0 ) ANG=60.0
C
C  Convert ang to radians
C
         ANGR = ANG / 180.0 * 3.141592
C
C  Set initial equal to true
```

```
C
         INITIAL=.TRUE.
C
         DO 100 J=1,NVEL
C
C  Set the velocity in ft/sec and km/sec
C
            VELM = J*VINCR
C
C  Convert vel to ft/sec
C
            VELE = VELM * 1.0E+05 / 2.54 / 12.0
C
C  Determine the critical diameter, as a function of wall configuration
C
            IF ( CTYPE.EQ.2 ) THEN
C
C  For the original and regression penetration functions use DOUBLE
C  subroutines
C
               IF ( PFUNC.EQ.1 .OR. PFUNC.EQ.3 ) THEN
                  CALL DOUBLE ( ANGR,DIA,INITIAL,ITYPE,MLI,PFUNC,SHTHK,
     1                           STAND,VELE,VELM,VWTHK,BHARD,C,DENS,
     2                           FSU,FTU,FY,SHPV,WILKC )
C
C  For Pen4 use the pen4 subroutine
C
               ELSE IF ( PFUNC.EQ.2 ) THEN
                  CALL PEN4  ( ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                        INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
               END IF
C
            ELSE
               CALL SINGLE ( ANGR,DIA,VELE,VWTHK,DENS,FTU )
            END IF
C
C  Convert the diameter to cm
C
            DIAM = DIA * 2.54
C
C  Store the diameter in RTABLE
C
            RTABLE(J,I,PID)=DIAM
C
 100     CONTINUE
C
 200  CONTINUE
C
      RETURN
C
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE DOUBLE (ANGR,DIA,INITIAL,ITYPE,MLI,PFUNC,SHTHK,STAND,
     .                    VELE,VELM,VWTHK,BHARD,C,DENS,FSU,FTU,FY,
     .                    SHPV,WILKC)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  DOUBLE DETERMINES THE PROJECTILE DIAMETER THAT JUST PENETRATES
C  THE GIVEN DOUBLE PLATE CONFIGURATION AT THE GIVEN IMPACT VELOCITY
C  AND ANGLE. IT IS USED FOR BOTH THE ORGINAL AND REGRESSION
C  PENETRATION FUNCTIONS.
C
C
C
C     VARIABLE LIST
C
C     ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL,RADIANS
C     BALL = LOGICAL PARAMETER USED TO DETERMINE IF THE BALLITIC
C            SUBROUTINES ARE CALLED
C     BHARD = ARRAY CONTAINING THE VALUES OF THE BRINNEL HARDNESS FOR
C             THE SHIELD & VESSEL WALL MATERIALS
C     BRIST = LOGICAL PARAMETER USED TO DETERMINE IF THE FRAGMENTING
C             SUBROUTINES ARE CALLED
C     C = ARRAY CONTAINING VALUES FOR THE SPEED OF SOUND (FT/SEC) FOR
C         THE SHIELD AND VESSEL WALL MATERIALS
C     DIA = PROJECTILE DIAMETER  ( IN. )
C     DIAB = DIAMETER AS DETERMINED BY SUBROUTINE BRISTOW ( IN. )
C     DIABL = DIAMETER AS DETERMINED BY SUBROUTINE BALLIST ( IN. )
C     DIAW = DIAMETER AS DETERMINED BY SUBROUTINE WILKIN  ( IN. )
C     EVWTHK = EQUIVALENT VESSEL WALL THK.  ( IN. )
C     FY = ARRAY CONTAINING THE VALUES OF THE YIELD STRENGTH FOR THE
C          SHIELD & VESSEL WALL MATERIALS
C     FSU = ARRAY CONTAING VALUES OF THE ULTIMATE SHEAR STRENGTH (PSI)
C           FOR THE SHIELD AND VESSEL WALL MATERIALS
C     FTU = ARRAY CONTAING VALUES OF THE ULTIMATE TENSILE STRENGTH (PSI)
C           FOR THE SHIELD AND VESSEL WALL MATERIALS
C     INITIAL = LOGICAL PARAMETER USED TO DETERMINE IF CURRENT
C               SUBROUTINE CALL IS INITIAL ONE
C     ITYPE = ANALYSIS TYPE 1=DEBRIS & METEOROIDS, 2=METEOROIDS
C     MLI = CHARACTER STRING USED TO DETERMINE IF MLI IS USED IN WALL
C     PFUNC = PENETRATION FUNCTION
C                1- ORIGINAL
C                2- PEN4
C                3- REGRESSION
C     SHATTER = LOGICAL PARAMETER USED TO DETERMINE IF PEN4 BALLISTIC
C               LIMITS HAVE BEEN EXCEEDED
C     SHPV = ARRAY CONTAINING THE VALUES FOR THE SHOCK PROJECTILE
C            VELOCITY (FT/SEC ) OF THE SHIELD & VESSEL WALL MATERIALS
C     SHTHK = SHIELD THICKNESS ( IN. )
C     STAND = SHIELD STAND-OFF DISTANCE (IN.)
C     THKMLI = EQUIVALENT THK OF 30 LAYERS OF MLI  ( IN. )
C     VELE = VEL IN FT/SEC
C     VELM = VEL IN KM/SEC
C     VWTHK = VESSEL WALL THICKNESS ( IN. )
C     WILKC = ARRAY  CONTAINING THE VALUES OF WILKINSON'S CONSTANT FOR
C             THE SHIELD & VESSEL WALL MATERIALS
C
      INTEGER*2 ITYPE,PFUNC
C
      DIMENSION  BHARD(3),C(3),DENS(3),FY(3),FSU(3),FTU(3),SHPV(3),
     .           WILKC(3)
C
      LOGICAL BALL,BRIST,INITIAL,MLI,SHATTER
C
C  INITIALIZE VARIABLES
C
      IF ( INITIAL ) THEN
         BALL=.TRUE.
         BRIST=.TRUE.
         DIA=.01
         SHATTER=.FALSE.
      END IF
C
C  TAKE 30 LAYERS OF MLI INTO ACCOUNT USING COUR-PALAIS'S EQN.
C  SET LIMIT FOR EQN. AT 10 KM/SEC  OR SKIP OVER SECTION ENTIRELY
C
C
      IF (.NOT. MLI ) THEN
         THKMLI = 0.0
         GO TO 50
      ENDIF
C
      IF ( VELM .LT. 10.0 ) THEN
         THKMLI = 3.045E-06 * ( VELM ** 3.42 )
      ELSE
         THKMLI = 3.045E-06 * ( 10.0 ** 3.42 )
      END IF
C
C  CONVERT TO IN.
C
      THKMLI = THKMLI / 2.54
C
C  ADD EQUIVALENT THK. OF MLI TO VESSEL WALL THK.
C
50    EVWTHK = VWTHK + THKMLI
C
C
C  DETERMINE DIAMETER IN IN. THAT  PENETRATES THE SHIELD AND
C  JUST DOES NOT PENETRATE THE VESSEL WALL
C
C  IF THE ANALYSIS IS FOR METEOROIDS, THEN ONLY USE WILKINSON'S METHOD
C  TO DETERMINE THE PENETRATION DIAMETER
C
C
      IF ( ITYPE .EQ. 2 .OR. VELM .GT. 12.0 )  THEN
          BALL=.FALSE.
          BRIST=.FALSE.
          GOTO 500
      ENDIF
C
C  INITIALLY CALCULATE THE PENETRATION DIAMETER USING BOTH THE
C  BALLISTIC AND FRAGMENTING SUBROUTINES. THE DIAMETER CALCULATED BY
C  THE BALLISTIC SUBROUTINE IS USED UNTIL THE VALUE CALCULATED FROM
C  THE FRAGMENTING SUBROUTINE IS GREATER. AT THAT TIME IT IS NO LONGER
C  NECESSARY TO CALL THE BALLISTIC SUBROUTINES.
C
C  FOR THE ORGINAL PENETRATION FUNCTION THE OLD VERSION OF PEN4 IS USED
C  IN THE BALLISTIC REGIME AND THE BURCH MODIFIED EQUATIONS ARE USED
C  IN THE FRAGMENTING REGIME.
C
C  FOR THE REGRESSION PENETRATION FUNCTION THE NEW PEN4 IS USED IN
C  THE BALLISTIC REGIME AND THE REGRESSION EQUATIONS ARE USED IN
C  FRAGMENTING REGIME.
```

```
C
C
      IF(BALL)THEN
        IF ( PFUNC.EQ.1 ) THEN
           CALL BALLIST (ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                  INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE)
        ELSE
           CALL PEN4 (  ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                  INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
        END IF
           DIABL=DIA
C
        IF ( PFUNC.EQ.1. ) THEN
           CALL BRISTOW (ANGR,C,DIA,EVWTHK,INITIAL,SHTHK,STAND,VELE)
        ELSE
           CALL REGRESS (  ANGR,DIA,MLI,SHTHK,STAND,VELM,VWTHK )
        END IF
        DIAB=DIA
C
C  CHECK IF THE DIAMETER CALCULATED BY BALLISTIC SUBROUTINE IS LESS
C  THAN THAT CALCULATED BY FRAGMENTING SUBROUTINE. IF SO SET BALL TO
C  FALSE .
C
        IF(DIAB.GT.DIABL)THEN
          BALL=.FALSE.
          DIA=DIAB
          GOTO 700
        ELSE
          BALL=.TRUE.
          DIA=DIABL
          GOTO 700
        ENDIF
C
C  CALCULATE THE PENETRATION DIAMETER USING BOTH THE FRAGMENTING AND
C  WILKIN SUBROUTINES. THE DIAMETR CALCULATED BY FRAGMENTING IS USED
C  UNTIL THE VALUE DETERMINED BY WILKIN IS LESS. IT IS THEN NOT
C  NECESSARY TO CALL FRAGMENTING.
C
C
      ELSE
        IF(BRIST)THEN
          IF ( PFUNC.EQ.1. ) THEN
             CALL BRISTOW (ANGR,C,DIA,EVWTHK,INITIAL,SHTHK,STAND,VELE)
          ELSE
             CALL REGRESS (  ANGR,DIA,MLI,SHTHK,STAND,VELM,VWTHK )
          END IF
          DIAB=DIA
C
          CALL WILKIN (ANGR,DENS,DIA,EVWTHK,ITYPE,SHTHK,STAND,VELM,
     1                 WILKC)
          DIAW=DIA
C
C  CHECK IF THE VALUE DETERMINED BY WILKIN IS LESS THEN THAT
C  DETERMINED BY BRISTOW. IF SO SET BRIST TO FALSE.
C
          IF(DIAW.LT.DIAB)THEN
            BRIST=.FALSE.
            DIA=DIAW
            GOTO 700
          ELSE
            BRIST=.TRUE.
            DIA=DIAB
            GOTO 700
           ENDIF
C
       ENDIF
C
      ENDIF
C
C
C  CALCULATE THE DIAMETER USING THE WILKIN SUBROUTINE.
C
500   CALL WILKIN (ANGR,DENS,DIA,EVWTHK,ITYPE,SHTHK,STAND,VELM,WILKC)
C
C  SET INITIAL TO FALSE
C
700   INITIAL = .FALSE.
C
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE BRISTOW  (ANGR,C,DIA,EVWTHK,INITIAL,SHTHK,STAND,VELE)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C     BRISTOW DETERMINES NUMBER OF PLATES PENETRATED. BASED ON
C     "MULTIPLE-DAMAGE STUDY", BY G.T. BURCH, BOEING 1967,AIR FORCE
C     ARMAMENT LABORATORY TECH REPORT AFATL-TR-67-116.
C
C     VALID FROM 3.6 TO 10.2 KM/SEC,AND FOR ALUMINUM PROJECTILES
C     IMPACTING ALUMINUM PLATES AT NORMAL .
C
C
C     VARIABLE LIST
C
C       ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL, RADIANS
C       C1 = SPEED OF SOUND IN SHIELD MATERIAL,FT/SEC
C       C = ARRAY CONTAINING VALUES OF THE SPEED OF SOUND FOR
C           THE SHIELD AND VESSEL WALL MATERIALS ( FT/SEC )
C       CHI = INTERMEDIATE VARIABLE
C       DIA = PROJECTILE DIAMETER,IN.
C       DIA1 = PREVIOUS DIAMETER
C       DIA2 = CURRENT DIAMETER
C       DT = INTERMEDIATE VARIABLE
C       EVWTHK = EQUIVALENT VESSEL WALL THICKNESS,IN.
C       F1 = INTERMEDIATE VARIABLE
C       F2 =          "           "
C       F3 =          "           "
C       I = ITERATION COUNTER
C       INITIAL = LOGICAL PARAMETER USED TO DETERMINE IF CURRENT CALL
C                 IS INITIAL ONE
C       NF = NUMBER OF PLATES PENETRATED BY THE FLIGHT PATH COMPONENT
C       NN = NUMBER OF PLATES PENETRATED BY THE NORMAL COMPONENT
C       PLP = # OF PLATES PENETRATED
C       PLP1 = PREVIOUS PLP
C       PLP2 = CURRENT PLP
C       SD = INTERMEDIATE VARIABLE
C       SHTHK = SHIELD THICKNESS,IN.
C       SIN3 = INTERMEDIATE VARIABLE
C       SLOPE = SLOPE OF LINE THROUGH (DIA1,PLP1) & (DIA2,PLP2)
C       SPF = NUMBER OF PLATES PENETRATED THAT ACCOUNTS FOR SPALLING
C       STAND = SHIELD STAND-OFF (IN.)
C       SWITCH = LOGICAL VARABLE USED TO DETERMINE IF THE CRITICAL
C                DIAMETER IS TO BE ESTIMATED THROUGH LINEAR
C                INTERPOLATION
C       T1D = INTERMEDIATE VARIABLE
C       T2D =          "
C       TEST = RATIO OF THE NUMBER OF PLATES PENETRATED TO THE
C              SPALLING FACTOR
C       VC =           "
C       VELE = VEL IN FT/SEC
C       VHOLD = VELOCITY HOLDER
C
C
C
      DIMENSION C(3)
C
      LOGICAL INITIAL
C
```

```
      REAL NF,NN
C
      LOGICAL SWITCH
C
      SAVE PLP1,PLP2,DIA1,DIA2
C
C
C  SINCE THIS METHOD DOES NOT SOLVE FOR THE DIAMETER THAT JUST
C  PENETRATES DIRECTLY AN ITERATIVE APPROACH IS TAKEN. INITIAL VALUES
C  ARE SET,AND USED TO APPROXIMATE THE CORRECT VALUE. THIS PROCESS
C  IS CONTINUED UNTIL THE ANSWER IS WITHIN TOLERANCES.
C
C
C  SET INITIAL VALUES
C
      I=0
      SPF = 0.85
      SWITCH=.FALSE.
C
C  IF THIS IS THE INITIAL CALL SET INITIAL  ELSE USE PREVIOUS VALUES
C  VALUES
C
      IF (INITIAL) THEN
         DIA2=10.0
      ENDIF
C
C   FOR VELOCITIES LESS THAN 11,800 FT/SEC SET THE VELOCITY EQUAL
C   TO 11,800 AND CALCULATE THE CRITICAL DIAMETER. USE THIS VALUE
C   TO ESTIMATE THE ACTUAL DIAMETER. SET THE LOGICAL VARIABLE SWITCH
C   TO TRUE AND SAVE THE VELOCITY AS VHOLD.
C
      IF ( VELE .LT. 11800. ) THEN
         VHOLD=VELE
         VELE=11800.
         SWITCH=.TRUE.
      END IF
C
C  CALCULATE INTERMEDIATE VARIABLES THAT DO NOT NEED TO BE CALCULATED
C  FOR EACH DIAMETER
C
      CHI=TAN(ANGR)-0.50
      SIN3=(SIN(ANGR))**3
      VC=VELE/C(1)
C
100   I=I+1
C
C  IF THIS IS THE FIRST PREDICTION USE THE DIAMETER VALUE THAT WAS
C  CALCULATED TO PENETRATE FOR THE PREVIOUS CASE AS A STARTING POINT.
C  ELSE USE A LINEAR PREDICTION APPROACH BASED  ON THE LAST TWO
C  PREDICTIONS.
C
      IF ( I.EQ.1 ) THEN
C
         DIA=DIA2
         DIA2=0.0
         PLP2=0.0
C
      ELSE
C
         SLOPE=(PLP2-PLP1)/(DIA2-DIA1)
         DIA=((SPF-PLP1)/SLOPE+DIA1)
C
C  CHECK THAT DIA > 0.0
```

```
C
         IF(DIA.LT.0.0) DIA=1.0E-10
C
C  Check if dia1 = dia2,if so stop
C
         IF ( DIA.EQ.DIA2 ) THEN
            WRITE ( 6,150 ) ANGR,VELE,DIA
  150       FORMAT (/1X,'BRISTOW CANNOT CONVERGE BECAUSE OF FLAT ',
     1              'SLOPE (ANGLE,VEL,DIA) =',3E12.5 )
            STOP
         END IF
      END IF
C
C  CALCULATE # OF PLATES PENETRATED
C
      SD=STAND/DIA
      T1D=SHTHK/DIA
      T2D=EVWTHK/DIA
C
      F1=2.42*(T1D**(-1./3.))+4.26*(T1D**(1./3.))-4.18
C
      F2=(0.5-1.87*T1D)+(5.*T1D-1.6)*(CHI**3)+(1.7-12.*T1D)*CHI
C
      F3=0.32*(T1D**(5./6.))+0.48*(T1D**(1./3.))*SIN3
C
      NF=(F1+0.63*F2)*(VC**(-4./3.))*(SD**(-5./12.))*(T2D**(-7./12.))
C
      NN=F3*(T2D**(-1.))*(VC**(-4./3.))
C
C  DETERMINE WHICH COMPONENT CONTROLS
C
      IF ( NF.GT.NN ) THEN
         PLP=NF
      ELSE
         PLP=NN
      ENDIF
C
C  RESET HOLDERS
C
      DIA1=DIA2
      DIA2=DIA
      PLP1=PLP2
      PLP2=PLP
C
C  CHECK IF PLP IS WITHIN TOLERANCE,IF NOT ITERATE
C
      TEST = PLP/SPF
C
C
      IF ( TEST.LT.0.99 .OR. TEST.GT.1.01 ) GO TO 100
C
C  IF SWITCH IS TRUE, ESTIMATE THE CRITICAL DIAMETER USING LINEAR
C  INTERPOLATION. THE TWO POINTS USED ARE THE ORIGIN AND THE VALUE
C  CALCULATED AT 11800 .
C
      IF ( SWITCH ) THEN
C
         DIA=DIA2*VHOLD/11800.0
         SWITCH=.FALSE.
         DIA1=0.0
         PLP1=0.0
C
      END IF
```

```
C
      RETURN
C
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE WILKIN (ANGR,DENS,DIA,EVWTHK,ITYPE,SHTHK,STAND,VELM,
     .                   WILKC )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  WILKIN DETERMINES THE DIAMETER THAT JUST PENETRATES THE VESSEL
C  WALL. IT IS BASED ON J.P.D. WILKINSON'S PAPER 'A PENETRATION
C  CRITERION FOR DOUBLED-WALLED STRUCTURE SUBJECT TO METEOROID IMPACT'
C  ,AIAA JOURNAL OCT 1969.
C
C  THE MAJOR ASSUMPTION USED BY WILKINSON IS THAT THE PROJECTILE
C  VAPORIZES ON IMPACT
C
C
C  VARIABLE LIST
C
C      ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL, RADIANS
C      CONST = MATERIAL CONSTANT DEFINED BY WILKINSON
C      C1 = INTERMEDIATE VARIABLE
C      DENS = ARRAY CONTAINING VALUES FOR DENSITY OF THE SHIELD &
C             VESSEL WALL MATERIALS
C      DIA = PROJECTILE DIAMETER (IN)
C      DIAM = PROJECTILE DIAMETER (CM )
C      EVWTHK = EQUIVALENT VESSEL WALL THK.  ( IN)
C      ITYPE = ANALYSIS TYPE  1=DEBRIS & METEOROIDS, 2=METEOROIDS
C      MASS = PROJECTILE MASS ( GRAMS )
C      MB = INTERMEDIATE VARIABLE
C      MT =       "          "
C      PI = PI
C      PROJDEN = PROJECTILE DENSITY ( G/CC )
C      RMI = UNIT MASS OF SHIELD (G/CM**2)
C      RM2 = UNIT MASS OF VESSEL WALL (G/CM**2)
C      SHDEN = SHIELD & VESSEL WALL DENISITY ( GRAMS/CC )
C      SHTHK = SHIELD THK. (IN)
C      STAND = SHIELD STAND-OFF DISTANCE  (IN)
C      STANDM = STAND IN CM.
C      VELM = VEL IN KM/SEC
C      VNORM = NORMAL COMPONENT OF THE VELEOCITY VECTOR,KM/SEC
C      VWDEN = VESSEL WALL DENSITY (LB/IN**3)
C      WILKC = ARRAY CONTAINING VALUES OF WILKINSON'S CONSTANT
C              FOR THE SHIELD & VESSEL WALL MATERIALS
C
C
C
      DIMENSION DENS(3),WILKC(3)
C
      INTEGER*2 ITYPE
C
      REAL MB,MT,MASS
C
C SET INITIAL VALUES
C
      PI = 3.141592654
C
C  SET PROJECTILE DENSITY IN G/CC
C
      PROJDEN = DENS(3)*27.705
C
C  SET SHIELD AND VESSEL WALL DENSITY IN G/CC
C
```

```
      SHDEN = DENS(1) * 27.705
      VWDEN = DENS(2) * 27.705
C
C  CONST IS A MATERIAL VARIABLE DEFINED IN THE PAPER & IS
C  EQUAL TO .401 FOR 2219 AL
C
      CONST = WILKC(2)
C
C  DETERMINE SHIELD & VESSEL WALL MASS PER UNIT AREA
C
      RM1 = SHTHK * 2.54 * SHDEN
      RM2 = EVWTHK * 2.54 * VWDEN
C
C  CONVERT STAND TO CM
C
      STANDM = STAND * 2.54
C
C  CALCULATE THE NORMAL COMPONENT OF THE VELOCITY VECTOR
C
      VNORM=VELM*COS(ANGR)
C
C  DETERMINE CRITICAL PROJECTILE DIAMETER
C
      MT = 1.44*(PI/6.0)**(1./3.)*CONST*RM1*RM2*STANDM**2.0
      MB = PROJDEN**(2./3.)*VNORM
      MASS = (MT/MB)**.75
      DIAM = (6.0*MASS/(PI*PROJDEN))**(1./3.)
C
C     CHECK IF APPROPRIATE EQ. WAS USED
C
      C1 = RM1 / ( PROJDEN * DIAM )
      IF ( C1 .GT. 1.0 ) THEN
C
C   WRONG EQN. USED,RECALC USING CORRECT EQN.
C
         MASS= 1.44*CONST*RM2*(STANDM**2.)/VNORM
         DIAM = (6.0*MASS/(PI*PROJDEN))**(1./3.)
      END IF
C
C  CONVERT DIAMETER TO IN
C
      DIA = DIAM / 2.54
C
      RETURN
C
      END
```

```
C
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
      SUBROUTINE REGRESS ( ANGR,DIA,MLI,SHTHK,STAND,VELM,VWTHK )
C
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C
C
C
C  Subroutine REGRESS determines the critical projectile diameter for
C  a two-plate structure impacted by aluminum spherical projectiles. It
C  was developed from test data obtained during the NASA contract
C  'Integrated Space Station Wall Design Guide and Penetration Control
C  Plan'. The data varied from 2 to 8 km/sec.
C
C
C  Variable List
C
C   angr = impact angle measured from the normal, radians
C   dia = critical projectile diameter, in
C   d13 = intermediate variable
C   high = holder, last diameter to penetrate , in
C   ic = counter
C   low = holder, last diameter to not penetrate, in
C   lsd = intermediate variable
C   mli = logical variable indicating wether multi-layer insulation is
C         used
C   mt = intermediate variable
C   np = number of plates penetrated, excluding the shield
C   shthk = shield thickness , in
C   stand = shield stand-off , in
C   switch = logical variable, used to determine if a penetrating
C            diameter has been determined
C   tt  = intermediate variable
C   t13 =      "         "
C   vc2 =      "         "
C   velm = impact velocity, km/sec
C   vwthk = vessel wall thickness , in
C
C
C
      INTEGER IC
C
      REAL LOW,LSD,MT,NP
C
      LOGICAL MLI,SWITCH
C
C  Intialize Variables
C
      IC=0
      LOW=0.0
      HIGH=5.0
      SWITCH=.TRUE.
C
C  Set MLI constant
C
      IF ( MLI ) THEN
         MT=-14.0
      ELSE
         MT=0.0
      END IF
C
C  Since the equation does not solve for the critical projectile
```

```
C  diameter directly, use a binary search technique to determine it.
C  First determine a diameter that penetrates then narrow in on the
C  actual diameter.
C
  100 CONTINUE
C
      IC=IC+1
C
      IF ( SWITCH ) THEN
         HIGH=HIGH*2.0
      END IF
C
      DIA=(HIGH+LOW)/2.0
C
C  Check that the diameter is less then 50 cm , if not stop
C
      IF ( DIA.GT.50.0 ) THEN
         WRITE ( 6,'(''---ERROR--- Diameter greater than 50 cm in '',
     1          ''subroutine REGRESS'')')
         STOP
      END IF
C
C  Calculate the intermediate variables
C
      D13=DIA**(1./3.)
      LSD=LOG10(STAND)/DIA
      T13=SHTHK**(1./3.)
      TT=TAN(ANGR)
      VC2=VELM*(COS(ANGR)**2)
C
C  Calculate the number of plates penetrated
C
      NP=1.52-6.18*T13-18.8*VWTHK-0.146*LSD+MT*SHTHK+10.8*D13-0.287*VC2
     1   -0.713*TT
C
C  Check for convergance
C
      IF ( IC.GT.100 ) THEN
         WRITE ( 6,'(''---ERROR--- REGRESS failed to converge after '',
     1          ''100 cycles'' )')
         STOP
      END IF
C
C  Has a diameter that penetrates been found, if not reset holders and
C  try again
C
      IF ( SWITCH ) THEN
         IF ( NP.GT.1 ) THEN
            SWITCH=.FALSE.
            HIGH=DIA
            GO TO 100
         END IF
      END IF
C
C  Does the diameter yeild an acceptable result, if not rest holders
C  and try again
C
      IF ( NP.LT.0.999 ) THEN
         LOW=DIA
         GO TO 100
      ELSE IF ( NP.GT.1.001 ) THEN
         HIGH=DIA
         GO TO 100
      END IF
C
C Finished
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE BALLIST (ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                    INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  BALLISTIC DETERMINES THE DIAMETER THAT JUST PENETRATES. IT ULTILZES
C  THE BALLISTIC PORTION OF BOEING'S HYPERVELOCITY CODE PEN4.
C
C
C  VARIABLE LIST
C
C    ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL,RADIANS
C    BHARD = ARRAY CONTAINING THE VALUES OF THE BRINNEL HARDNESS FOR
C            THE SHIELD & VESSEL WALL MATERIALS
C    C = ARRAY CONTAINING VALUES FOR THE SPEED OF SOUND FOR THE
C        SHIELD AND VESSEL WALL MATERIALS (FT/SEC)
C    DEN = PROJECTILE DENSITY (LB/CFT)
C    DENS = ARRAY CONTAINING VALUES FOR THE DENSITY FOR THE
C           SHIELD AND VESSEL WALL MATERIALS (LB/IN**3)
C    DIA = SPHERICAL PROJECTILE DIAMETER (IN)
C    DIA1 = PREVIOUS DIAMETER THAT PENETRATED (IN)
C    DIA2 = DIAMETER THAT PENETRATED TWO ITERATIONS PREVIOUSLY (IN)
C    EVWTHK = EQUIVALENT VESSEL WALL THICKNESS,IN.
C    FY = ARRAY CONTAINING THE VALUES OF THE YIELD STRENGTH FOR THE
C         SHIELD & VESSEL WALL MATERIALS
C    FSU = ARRAY CONTAINING VALUES OF THE ULTIMATE SHEAR STRENGTH
C          FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C    FTU = ARRAY CONTAINING VALUES OF THE ULTIMATE TENSILE STRENGTH
C          FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C    I = COUNTER
C    INITIAL = LOGICAL PARAMETER USED TO DETRMINE IF CURRENT CALL
C              IS INITIAL ONE.
C    MASS = PROJECTILE MASS,LBS
C    N = INCREMENT MULIPLIER
C    PDENS = DENS ARRAY CONVERTED TO SLUGS/FT**3
C    PFY = FY ARRAY CONVERTED TO PSF
C    PFSU = FSU ARRAY CONVERTED TO PSF
C    PFTU = FTU ARRAY CONVERTED TO PSF
C    P1 = LAST MASS GUESS TO NOT PENETRATE
C    P2 = LAST MASS GUESS TO PENETRATE
C    PEN = TRUE OR FALSE
C    PI = 3.14
C    PMINCR = INITIAL MASS GUESS INCREMENT
C    RATIO = P2 / P1
C    SHATTER = LOGICAL PARAMETER USED TO DETERMINE IF PEN4 BALLISTIC
C              LIMITS HAVE BEEN EXCEEDED
C    SHPV = ARRAY CONTAING VALUES OF THE SHOCK PROJECTILE VELOCITY
C           FOR THE SHIELD AND VESSEL WALL MATERIALS, ( UNITLESS )
C    SHTHK = SHIELD THICKNESS,IN.
C    SPACE = ARRAY CONTAINING THE SHIELD SPACING ,FT.
C    STAND = SHIELD STAND-OFF,IN.
C    TARMAT = ARRAY CONTAINING MATERIAL PROPERTY POINTERS
C    THETA = IMPACT ANGLE (RAD),MEASURED FROM THE NORMAL
C    THICK = SHIELD & VESSEL WALL THICKNESS,FT.
C    VELE = COLLISION VELOCITY,FT/SEC
C    VEL1 = VEL FOR DIA1
C    VEL2 = VEL FOR DIA2
C    V0 = IMPACT VELOCITY (FT/SEC)
C    VR2 = RESIDUAL VELOCITY AFTER VESSEL WALL,FT/SEC
C
```

```
C
C
      DIMENSION BHARD(3),C(3),DENS(3),FY(3),FSU(3),FTU(3),PDENS(3),
     .          PFY(3),PFSU(3),PFTU(3),SHPV(3),SPACE(2),THICK(3),
     .          TARMAT(3)
C
      INTEGER PROJMAT,TARMAT
C
      LOGICAL PEN,INITIAL,SHATTER
C
      REAL MASS
C
      SAVE DIA1,DIA2,P2,VEL1,VEL2
C
C
C  SET INITIAL VALUES
C
      I = 0
      N = 0
      PEN = .FALSE.
      PMINCR = 1.0E-04
      PI = 3.1415926536
      P1 = 1.0E-06
C
C  CONVERT VARIABLES TO PEN4 FORMAT AND UNITS
C
      DO 50 I=1,3
C
C  CONVERT DENS TO SLUGS/FT**3
C
         PDENS(I)=DENS(I)*1728./32.2
C
C  CONVERT FSU AND FTU TO PSF
C
         PFY(I)=FY(I)*144.
         PFTU(I)=FTU(I)*144.
         PFSU(I)=FSU(I)*144.
C
50    CONTINUE
C
      PROJMAT = 3
      SPACE(1) = STAND/12.0
      SPACE(2) = 0.0
      TARMAT(1) = 1
      TARMAT(2) = 2
      TARMAT(3) = 0
      THETA = ANGR
      THICK(1) = SHTHK/12.0
      THICK(2) = VWTHK/12.0
      THICK(3) = 0.0
      V0 = VELE
C
C  PEN4 DOES NOT SOLVE FOR THE CRITICAL PROJECTILE DIAMETER DIRECTLY
C  THEREFORE AN ITERATIVE APPROACH IS TAKEN. PEN4 ONLY RETURNS
C  A LOGICAL PARAMETER INDICATING IF PENETRATION HAS OCCURED.
C
C
C  FOR INITIAL CASE SET P2 > 0.0
C
      IF(INITIAL)THEN
         P2 = 1.0E-04
      ENDIF
C
```

```
C  IF SHATTER HAS OCCURED NO NEED TO CALL PEN4,USE LINEAR APPROX.
C
      IF (SHATTER) GO TO 500
C
C  DETERMINE INITIAL MASS THAT PENETRATES
C
100   MASS = P2 + N * PMINCR
C
      NC = NC + 1
C
      CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,PROJMAT,SHPV,
     .          C,SPACE,TARMAT,THICK,PFY,*500)
C
      IF ( PEN ) THEN
C
C    IF PENETRATION TRY WITH MASS BETWEEN P1 & P2
C
         PEN = .FALSE.
         P2 = MASS
         MASS = ( P1 + P2 ) / 2.0
C
         NC = NC + 1
         CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,PROJMAT,
     .               SHPV,C,SPACE,TARMAT,THICK,PFY,*500)
C
C
200      IF ( PEN ) THEN
C
C      IF PENETRATION,SET P2 = MASS & CHECK RATIO
C
            PEN = .FALSE.
            P2 = MASS
            RATIO = P2 /P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,
     .                     PROJMAT,SHPV,C,SPACE,TARMAT,THICK,PFY,
     .                     *500)
               GO TO 200
            ELSE
C
C        IF FALSE,MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
C
         ELSE
C
C      IF FALSE,SET P1 = MASS AND CHECK RATIO
C
            P1 = MASS
            RATIO = P2 / P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE,TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL OPEN4 (PEN,MASS,THETA,SHATTER,V0,PDENS,BHARD,
     .                    PROJMAT,SHPV,C,SPACE,TARMAT,THICK,PFY,
     .                    *500)
               GO TO 200
            ELSE
C
C        IF FALSE,MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
         ENDIF
      ELSE
C
C    IF FALSE INCREASE INCREMENT & TRY AGAIN
C
         P1 = MASS
         I = I + 1
         N = 2 ** I
         GO TO 100
      END IF
C
      P2=MASS
C
C  CALCULATE DIAMETER
C
C
C
300   DEN = DENS(3)*1728.0
C
      DIA = ((MASS*6.0)/(PI*DEN))**(1.0/3.0)
C
C  CONVERT TO IN.
C
      DIA = DIA * 12.0
C
C
C
C  RESET HOLDERS
C
      DIA2=DIA1
      DIA1=DIA
      VEL2=VEL1
      VEL1=VELE
C
C
      RETURN
C
C  SINCE SHATTER HAS OCCURED NO NEED TO CALL PEN4 USE LINEAR APPROXL
C
500   SLOPE=(DIA1-DIA2)/(VEL1-VEL2)
      DIA=DIA2+SLOPE*(VELE-VEL2)
      RETURN
C
      END
```

```
C
      SUBROUTINE OPEN4 (Penetration,Mass,Theta,SHATTER,V0,
     1                 Density,Hardness,ProjMat,ShockProjVel,SoundVel,
     2                 Spacing,TarMat,Thick,YieldStrength,*)
C     This version of pen4 is strictly for use in the sub-shatter velocity
C       regime.  It is a modified version of spin14, created on 6/21/85.

      LOGICAL Penetration,SHATTER
      REAL Mass,LAMBDA,NR,NF,NH,J,NR2,NHT,LastPK,MR
      INTEGER ProjMat,TarMat,Exponent,TopCount,BottomCount1,BottomCount2
      DIMENSION Thick(3),TarMat(3),Spacing(2),
     :          Density(3),YieldStrength(3),SoundVel(3),
     :              ShockProjVel(3),Hardness(3),Epsil(2),
     :          NR(3),NR2(3),RF(3),Flagit(3)
C
C          Density in Slugs/CubicFoot
C          YieldStrength in Lbs/SquareFoot
C          SoundVel in Feet/Second
C          Hardness is Brinell Scale
      DATA Epsil,Gamma/5.71,5.71,90.0/
      DATA F1,F2/4.0,1.0/
      DATA EffectiveThicknessRatio/.6/
      DATA A,B/2.0,3.125E-04/
      PI=3.141592653589793
      Small = 1E-36
      RecipSqrt2PI=1./SQRT(2.*PI)
C     **************Calculate Radius of Projectile Sphere*************
      RP=(Mass*3./
     1                 (Density(ProjMat)*32.2*4.*PI))**(1./3.)
      Diam = RP * 2.
C     ****************************************************************

C     ************Compute ResidualVel**********************************
      VDEL=( V0**2*1.33*RP**2*Density(ProjMat)-
     1 F1*YieldStrength(TarMat(1))*THICK(1)**2/COS(THETA)**2
     2 *A*EXP(-B*V0) )
     2 /( 1.33*RP**2*Density(ProjMat)+F2*RP*THICK(1)*
     3 Density(TarMat(1))/COS(THETA) )
      VR=SQRT(AMAX1(VDEL,0.))
C     ****************************************************************

C     ************Cratering V50****************************************
      V50=SQRT((Thick(1)*EffectiveThicknessRatio/COS(Theta)/
     1  (0.281*Diam*(Density(ProjMat)/Density(TarMat(1)))**(1./3.)))
     2  **(1./.31)/Density(ProjMat)*(2.*YieldStrength(TarMat(1))))
C     ****************************************************************

C     **************IF No Penetration Report Result and Exit*********
      IF (V0.LT.V50) THEN
            Penetration = .FALSE.
            RETURN
      ELSE IF (Thick(2).EQ.0.0) THEN
            Penetration = .TRUE.
            RETURN
      ENDIF
C     ****************************************************************

C     **************Mass Loss Regime Decision************************
      ToverD = Thick(1)/Diam
      FTHETA=-2.433E-4 - 1.643E-2*COS(THETA)+3.201*COS(THETA)**2-
     :    2.184*COS(THETA)**3
      IF (ToverD.LT..40) THEN
            Vf = 4100.
      ELSE
            Vf = 4986.*ToverD**.21
      END IF
      IF (V0.LT.VF+4000.) THEN
        IF (V0.LT.Vf) THEN
              CALL MassErr(V0,Mass,Hardness(ProjMat),
     :         Density(ProjMat),Density(TarMat(1)),Thick(1),Diam,MR)
            ELSE
              CALL Fract(MR,V0,Vf,FTheta,Mass,ToverD)
            END IF
            CALL LarMR(VR,MR,V50,Thick(2),EffectiveThickNessRatio,Theta,
     :                Diam,Density(ProjMat),Density(TarMat(2)),
     :                YieldStrength(TarMat(1)),Penetration)
      ELSE
C       **********The rest of this subroutine contains the evaluation******
C       *********** of shatter regime multiple cratering penetration*******
C       ######Shatter regime removed for independent use of ballistic######
C       ################And fracture regime evaluation#####################
        SHATTER = .TRUE.
            RETURN 1
      END IF
      RETURN
      END

      SUBROUTINE Fract(MR,V0,Vf,FTheta,MProj,ToverD)
      REAL MR,MProj
      IF (V0.GT.Vf+2000.) THEN
            MR = MProj*FTheta*.16
      ELSE IF (ToverD.GT..25) THEN
            MR = MProj*FTheta*.25
      ELSE
            MR = MProj*FTheta*.667
      END IF
      RETURN
      END

      SUBROUTINE LarMR(VR,MR,V50,Thickness,
     :                 EffectiveThicknessRatio,Theta,Diam,ProjDensity,
     :                 TargetDensity,TargetYieldStrength,Penetration)
      REAL MR
      LOGICAL Penetration
      V50=SQRT((Thickness*EffectiveThicknessRatio/COS(Theta)/
     1  (0.281*Diam*(ProjDensity/TargetDensity)**.33333))
     2  **(1./.31)/ProjDensity*(2.*TargetYieldStrength))
      IF (VR.LT.V50) THEN
            Penetration = .FALSE.
      ELSE
            Penetration = .TRUE.
      END IF
      RETURN
      END

      SUBROUTINE MassErr(V1,MFC1,BHNC,RhoP,RhoT,Thick,Diam,MR)
      REAL MR,MS1,MFC1
      RhoC = RhoP*.01873
      RhoS = RhoT*.01873
      T = Thick*12.
      DFC1 = Diam*12
C     DFC1 = SQRT(4*AC/PI)
      UFC=SQRT(2680*BHNC/RHOC)
      VP=V1/UFC
      AC1=PI*DFC1**2/(4*COS(THETA))
      MS1=RHOS*T*AC1
      MR=MFC1+0.5*MS1*LOG(2.74/(1+VP**2))
      MR = AMIN1(MR,MFC1)
      RETURN
      END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE SINGLE ( ANGR,DIA,VELE,VWTHK,DENS,FTU )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Single determines the critical projectile diameter for single plate
C  configurations. It is based on the Schmidt-Holsapple crater volume
C  equation. The equation was solved for the critical diameter as a
C  function of the plate and projectile properties in english units.
C  It assumes that a hemispherical crater depth equal to 70 percent of
C  the plate thickness causes failure. This is a attempt to account for
C  spall and was reccomended by Mike Bjorkman of the Boeing SHock Physics
C  group.
C
C
C  Ref: 'On Scaling of the Crater Dimensions 2. Impact Processes',
C       K.A. Holsapple & R.M. Schmidt, JGeophy Res, v87,nb3,10 March 1982,
C       p1849-70
C
C
C  Variable List
C
C    angr = impact angle measured from the normal, radians
C    dia = critical projectile diameter, in
C    dr = intermediate variable
C    fr =     "            "

C    vele = relative velocity of the projectile, ft/sec
C    vwthk = vessel wall thickness, in
C
C  Array list
C
C    dens = density, lb/in**3
C              1- shield
C              2- vessel wall
C              3- projectile
C    ftu = ultimate tensile strength, psi
C              1- shield
C              2- vessel wall
C              3- projectile
C
C
      DIMENSION DENS(3),FTU(3)
C
      DR=(DENS(3)/DENS(2))**(-0.159)
      V2=(VELE*COS(ANGR))**2
      FR=(2.6833*FTU(2)/DENS(3)/V2)**0.236
C
      DIA=1.442*DR*FR*VWTHK
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE PEN4 ( ANGR,BHARD,C,DENS,DIA,VWTHK,FY,FSU,FTU,
     1                   INITIAL,SHATTER,SHPV,SHTHK,STAND,VELE )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C  PEN4 DETERMINES THE DIAMETER THAT JUST PENETRATES. IT ULTILZES
C  THE UPDATED  BOEING'S HYPERVELOCITY CODE PEN4 VERSION 10.
C
C
C  VARIABLE LIST
C
C    ANGR = IMPACT ANGLE MEASURED FROM THE NORMAL , RADIANS
C    BHARD = ARRAY CONTAINING THE VALUES OF THE BRINNEL HARDNESS FOR
C            THE SHIELD & VESSEL WALL MATERIALS
C    C = ARRAY CONTAINING VALUES FOR THE SPEED OF SOUND FOR THE
C        SHIELD AND VESSEL WALL MATERIALS (FT/SEC)
C    DEN = PROJECTILE DENSITY (LB/CFT)
C    DENS = ARRAY CONTAINING VALUES FOR THE DENSITY FOR THE
C           SHIELD AND VESSEL WALL MATERIALS (LB/IN**3)
C    DIA = SPHERICAL PROJECTILE DIAMETER (IN)
C    DIA1 = PREVIOUS DIAMETER THAT PENETRATED (IN)
C    DIA2 = DIAMETER THAT PENETRATED TWO ITERATIONS PREVIOUSLY (IN)
C    EVWTHK = EQUIVALENT VESSEL WALL THICKNESS , IN.
C    FY = ARRAY CONTAINING THE VALUES OF THE YIELD STRENGTH FOR THE
C         SHIELD & VESSEL WALL MATERIALS
C    FSU = ARRAY CONTAINING VALUES OF THE ULTIMATE SHEAR STRENGTH
C          FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C    FTU = ARRAY CONTAINING VALUES OF THE ULTIMATE TENSILE STRENGTH
C          FOR THE SHIELD AND VESSEL WALL MATERIALS (PSI)
C    I = COUNTER
C    INITIAL = LOGICAL PARAMETER USED TO DETRMINE IF CURRENT CALL
C              IS INITIAL ONE.
C    MASS = PROJECTILE MASS , LBS
C    N = INCREMENT MULIPLIER
C    P1 = LAST MASS GUESS TO NOT PENETRATE
C    P2 = LAST MASS GUESS TO PENETRATE
C    PEN = TRUE OR FALSE
C    PI = 3.14
C    PMINCR = INITIAL MASS GUESS INCREMENT
C    RATIO = P2 / P1
C    SHATTER = LOGICAL PARAMETER USED TO DETERMINE IF PEN4 BALLISTIC
C              LIMITS HAVE BEEN EXCEEDED
C    SHPV = ARRAY CONTAING VALUES OF THE SHOCK PROJECTILE VELOCITY
C           FOR THE SHIELD AND VESSEL WALL MATERIALS , ( UNITLESS )
C    SHTHK = SHIELD THICKNESS , IN.
C    SPACE = ARRAY CONTAINING THE SHIELD SPACING ,FT.
C    STAND = SHIELD STAND-OFF , IN.
C    TARMAT = ARRAY CONTAINING MATERIAL PROPERTY POINTERS
C    THETA = IMPACT ANGLE (DEG) , MEASURED FROM THE NORMAL
C    THICK = SHIELD & VESSEL WALL THICKNESS , FT.
C    VELE = COLLISION VELOCITY , FT/SEC
C    VEL1 = VEL FOR DIA1
C    VEL2 = VEL FOR DIA2
C    V0 = IMPACT VELOCITY (FT/SEC)
C    VR2 = RESIDUAL VELOCITY AFTER VESSEL WALL , FT/SEC
C
C
C
      DIMENSION BHARD(3),C(3),DENS(3),FY(3),FSU(3),FTU(3),PDENS(3),
     .          PFY(3),PFSU(3),PFTU(3),SHPV(3),SPACE(2),THICK(3),
     .              TARMAT(3)
C
      INTEGER PROJMAT,TARMAT
C
      LOGICAL PEN,INITIAL,SHATTER
C
      REAL MASS
C
      SAVE DIA1,DIA2,P2,VEL1,VEL2
C
C
C  SET INITIAL VALUES
C
      I = 0
      N = 0
      PEN = .FALSE.
      PMINCR = 1.0E-04
      PI = 3.1415926536
      P1 = 1.0E-06
C
C  CONVERT VARIABLES TO PEN4 FORMAT AND UNITS
C
C
      PROJMAT=2
      SPACE(1) = STAND/12.0
      SPACE(2) = 0.0
      TARMAT(1) = 1
      TARMAT(2) = 2
      TARMAT(3) = 0
      THETA = ANGR*180.0/PI
      THICK(1) = SHTHK/12.0
      THICK(2) = VWTHK/12.0
      THICK(3) = 0.0
C
C  PEN4 DOES NOT SOLVE FOR THE CRITICAL PROJECTILE DIAMETER DIRECTLY
C  THEREFORE AN ITERATIVE APPROACH IS TAKEN. PEN4 ONLY RETURNS
C  A LOGICAL PARAMETER INDICATING IF PENETRATION HAS OCCURED.
C
C
C  FOR INITIAL CASE SET P2 > 0.0
C
      IF(INITIAL)THEN
         P2 = 1.0E-04
      ENDIF
C
C  IF SHATTER HAS OCCURED NO NEED TO CALL PEN4 , USE LINEAR APPROX.
C
C
C  DETERMINE INITIAL MASS THAT PENETRATES
C
100   MASS = P2 + N * PMINCR
C
      NC = NC + 1
C
      CALL NPEN4 ( VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,PROJMAT,
     1             DENS,FTU,C )
C
      IF ( PEN ) THEN
C
C    IF PENETRATION TRY WITH MASS BETWEEN P1 & P2
C
         PEN = .FALSE.
         P2 = MASS
         MASS = ( P1 + P2 ) / 2.0
C
         NC = NC + 1
         CALL NPEN4 ( VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,PROJMAT,
     1               DENS,FTU,C )
C
C
200      IF ( PEN ) THEN
C
C      IF PENETRATION , SET P2 = MASS & CHECK RATIO
C
            PEN = .FALSE.
            P2 = MASS
            RATIO = P2 /P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL NPEN4 (VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,
     1                    PROJMAT,DENS,FTU,C )
               GO TO 200
            ELSE
C
C        IF FALSE , MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
C
         ELSE
C
C      IF FALSE , SET P1 = MASS AND CHECK RATIO
C
            P1 = MASS
            RATIO = P2 / P1
            IF ( RATIO .LT. .995 .OR. RATIO .GT. 1.005 ) THEN
C
C        IF TRUE , TRY AGAIN BETWEEN P1 & P2
C
               MASS = ( P1 + P2 ) / 2.0
               NC = NC + 1
               CALL NPEN4 (VELE,MASS,THETA,THICK,SPACE,PEN,SHATTER,
     1                    PROJMAT,DENS,FTU,C)
               GO TO 200
            ELSE
C
C        IF FALSE , MASS IS WITHIN RANGE SO CONTINUE
C
               GO TO 300
            END IF
         ENDIF
      ELSE
C
C    IF FALSE INCREASE INCREMENT & TRY AGAIN
C
         P1 = MASS
         I = I + 1
         N = 2 ** I
         GO TO 100
      END IF
C
      P2=MASS
```

```
C
C  CALCULATE DIAMETER
C
C
C
300   DEN = DENS(3)*1728.0
C
      DIA = ((MASS*6.0)/(PI*DEN))**(1.0/3.0)
C
C  CONVERT TO IN.
C
      DIA = DIA * 12.0
C
C
C
C  RESET HOLDERS
C
      DIA2=DIA1
      DIA1=DIA
      VEL2=VEL1
      VEL1=VELE
C
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE NPEN4 (Vil,MProj,Thetal,Thickl,Space,Pennon,
     :       Shater,PrMatl,Densel,YStrnl,SoundV)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C
C     PrMat Integer Projectile Material 1:Steel, 2:Aluminum ,3:Ice

      CHARACTER Shape*3
      LOGICAL Pennon,Shater
      INTEGER Maxk(5)
      INTEGER PrMat,PrMatl,TarMat(10),TMatSp(10),Plate,Bin,NBin,I
      REAL RF(5),RC(5)
      REAL NF,J,MR,MProj,LastSp,LRM,FrMass(5)
      REAL Diam,Vi,Vr,ViLRM,VrLRM,Epsil,Gamma,Vil,Pi,Theta,SumSp
      REAL A,B,D,R,X,Y,ToverD,Rh,PlateM,FTheta,AllMas,Vc,DelJ,DelJ2
      REAL P,EffP,Pet,Pgrady,Thetal,AvgMas,Rp,F1,Vf
      REAL      Thick(10),Space(9),Thickl(10),
     :          PDense(3),PYStrn(3),PSondV(3),FrTuff(3),
     :          Dense(10),YStren(10),SoundV(10)
      REAL Densel(10),YStrnl(10)
      REAL ViX,MrMax,MProjX
      DOUBLE PRECISION Intact,HoArea
      DOUBLE PRECISION SumPr(5)
      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk,As,Ac

      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)
      COMMON /Crater/As,Ac(5),P(5)
C                                    Steel   Aluminum   Ice
      DATA PDense/                   15.11 ,      5.39 ,   1.94/
      DATA PYStrn/                8.35E+07 , 5485000.,   0.0/
      DATA PSondV/                  14960. ,   17569.,   0.0/
      DATA FrTuff/                     36. ,       39.,   39./
C       PrMat  Projectile Material 1:Steel, 2:Aluminum ,3:Ice
C       Calculation units
C          Dense  in Slugs/CubicFoot       Density
C          YStren in Lbs/SquareFoot        Uniaxial Ultimate Stress
C          SoundV in Feet/Second           Speed of Sound
C          Theta  in Radians               Impact Angle
C          FrTuff in MegaPascals Meter^.5  Fracture Toughness
C       Input units
C          Densel  input in Lbs/CubicInch
C          YStrnl  input in PSI
C          SoundV  input in Feet/Sec
C          Vil     Feet/Sec             Impact Velocity
C          MProj   Pounds               Projectile Mass
C          Thetal  Degrees              Striking Angle
C          Thick   Feet                 Target Plate Thickness
C          Space   Feet                 Spacing Between Target Plates
C        Outputs
C          Pennon Logical              Penetration Flag
C          Shater Logical              Fragment Shatter Flag

      DATA Gamma/1.5708/
      DATA F1/4.0/
      DATA NBin/5/
      DATA TarMat/1,2,3,4,5,6,7,8,9,10/
      DATA TMatSp/2,2,2,2,2,2,2,2,2,2/
      DO 10 Plate=1,10
```

```
         Thick(Plate)=Thick1(Plate)
         Dense(Plate)=Dense1(Plate)/32.2*1728.
         YStren(Plate)=YStrn1(Plate)*144.
  10   CONTINUE
       Theta = Theta1/57.3
       PrMat = PrMat1
C      END DO
       Vi=Vi1
       PI=3.141592653589793
       Shater =.FALSE.
C      ***************Calculate Radius of Projectile Sphere*************
       RP=(MProj*3./(PDense(PrMat)*32.2*4.*PI))**(1./3.)
       Diam = RP * 2.
C      *****************************************************************

       DO 2040 Plate=1,10
         IF (.NOT.Shater) THEN
C          *** This option is for single penetrators*************************
C          **Compute Residual Velocity

           Call ResVel
     :            (Vr,Vi,RP,PDense(PrMat),Dense(TarMat(Plate)),
     :       Thick(Plate),Theta,PrMat,TMatSp(Plate))

C          ***************IF No Penetration Report Result and Exit**********
           IF (Vr.EQ.0.) THEN
             Pennon = .FALSE.
             RETURN
           ELSE IF (Thick(Plate+1).EQ.0.0.OR.Plate.EQ.10) THEN
             Pennon = .TRUE.
             RETURN
           ENDIF
C          *****************************************************************
C          **************Mass Loss Regime Decision BAC IR&D****************
           ToverD = Thick(Plate)/Diam
           IF (PrMat.EQ.2.AND.TMatSp(Plate).EQ.2) THEN
             IF (ToverD.LT..1) THEN
               Vf = 1116*ToverD**(-.55)
             ELSE
               Vf = 4757*ToverD**.08
             END IF
           ELSE IF (PrMat.EQ.1.AND.TMatSp(Plate).EQ.2.) THEN
             IF (Shape.EQ.'CYL') THEN
               Vf=5020* ToverD**.4
             ELSE IF (Shape.EQ.'CUB') THEN
               Vf = AMAX1(1450*ToverD**(-.39),
     :                  4561*ToverD**42*(Diam*12*2.54)**(-.33))
             ELSE
               Vf = AMAX1(1450*ToverD**(-.39),
     :                  4561*ToverD**.42*(Diam*12*2.54)**(-.33) )

C               Vf = AMAX1(2362*ToverD**(-.35)*(Diam*12*2.54)**(-.32),
C    :                   3937*ToverD**.23   *(Diam*12*2.54)**(-.25))
             END IF
           ELSE IF (PrMat.EQ.1.AND.TMatSp(Plate).EQ.1.) THEN
             Vf = 7021*ToverD**.39
           END IF
C          ***************************************************************
C           RH=RP*(1.372E-4*Vi*(THICK(Plate)/(2.*RP))**(2./3.)+.9)*
C    :        (1-EXP(-(1.48-Theta)/.166))
           RH = .5*Thick(Plate)*11.02*(1-EXP(-(1.48-Theta)/.166))*
     :       (1-EXP(-(PDense(PrMat)*Vi**2/YStren(TarMat(Plate)))**.415
     :       *(PDense(PrMat)/Dense(TarMat(Plate)))**(-.15)/ToverD/29.9))
```

```
          RH = AMAX1(RH,RP)
          IF (Vi.GT.Vf) THEN
            SumSp=Space(Plate)
            Shater=.TRUE.
C           *** Compute plate spall ******************************************
            PlateM=PI*RH**2/COS(Theta)*Thick(Plate)*
     :                  Dense(TarMat(Plate))*32.2
C           *****************************************************************
C           *********Hydrocode Predicted Mass Loss Due to Impact Angle******
            FTHETA=-2.433E-4 - 1.643E-2*COS(THETA)+3.201*COS(THETA)**2-
     :        2.184*COS(THETA)**3
C           *****************************************************************
C           *** COMPUTE FRAGMENT NUMBERS*************************************
            IF (PrMat.EQ.3) THEN
              MProj=PlateM
              RP=(PlateM*3./(Dense(TarMat(Plate))*32.2*4.*PI))**(1./3.)
            END IF

            CALL MasChr (Vi,MProj,ToverD,Theta,RP,PDense(PrMat),
     :                   Dense(TarMat(Plate)),PSondV(PrMat),NBin,
     :                   RF,NR,LRM,AvgMas,PrMat,FrTuff(PrMat))

            IF (PrMat.EQ.3) THEN
              PrMat=TMatSp(Plate)
            ELSE
              AllMas=PlateM+MProj
              DO 2010 I=1,NBin-1
                FrMass(I)=4./3.*PI*RF(I)**3*PDense(PrMat)*32.2
                NR(I)=NR(I)*AllMas/MProj
 2010         CONTINUE
            END IF
            Nr(NBin) = 1.D0
            ViLRM=Vr
C           ***********Calculate Spray Angle*********************************
C            Vc =11155*ToverD**(-.52)
C            Epsil = 45*(1-EXP((-(Vi-Vf)/Vc)))/57.3
            Vc = 4889 * ToverD**(-.23)
            Epsil = 52*(1-EXP((-(Vi-Vf)/Vc)))/57.3
C           *** Assurance of ThetaR+Epsil<90 and Spray Area finite***********
            ThetaR = AMIN1(Theta,1.41-Epsil)
C           *** Calculate Spray Area*****************************************
            DELJ=RP/2.0*(COS(EPSIL)-TAN(ThetaR)*SIN(EPSIL))
            DELJ2=RP*(1.0-TAN(ThetaR)*TAN(EPSIL))
            J=Space(Plate)*SIN(GAMMA)/SIN(ThetaR+GAMMA)
            X=(J*SIN(EPSIL)/2.+DELJ)*(1./SIN(ThetaR+GAMMA+EPSIL)-
     :        1./SIN(ThetaR+GAMMA-EPSIL))
            Y=X*SIN(ThetaR+GAMMA)
            R=(J-X*COS(ThetaR+GAMMA))*TAN(EPSIL)+DELJ2
            B=SQRT(R**2-Y**2)
            A=(J*SIN(EPSIL)/2.+DELJ)*(1./SIN(ThetaR+GAMMA-
     :        EPSIL)+1./SIN(ThetaR+GAMMA+EPSIL))
            AS=PI*A*B
C           ********************************************************
C           *** Allowance for increased penetration due to spalling of ****
C           *** next plate *********************************************
              EffP=1.7
C           *****************************************************************
          ELSE
            IF (PrMat.EQ.3) THEN
              MProj=3.1415926*Rh**2*Thick(Plate)
     :                     *Dense(TarMat(Plate))*32.2
              PrMat=TMatSp(Plate)
            END IF
            Vi=Vr
          END IF
C         **************************************************************
        ELSE

C         **********The rest of this subroutine contains the evaluation*
C         *********** of shatter and fracture regime multiple cratering*

C         *** Allowance for increased penetration due to spalling of ****
C         *** next plate ************************************************
          Thick(Plate) = Thick(Plate)/EffP
C         ***************************************************************

C         *** COMPUTE PLATE CRATER Depth and Radius**********************
          DO 2030 I=1,NBin
            D=2.*RF(I)
            PGrady=0.281*D*(PDense(PrMat)/Dense(TarMat(Plate)))**(1./3.)
     :        *(PDense(PrMat)*(Vr*COS(Theta))**2/
     :        (2.*YStren(TarMat(Plate))))**0.31
            IF (PrMat.EQ.1) THEN
              Pet = D*(Vr*COS(Theta)/12468.)**1.32
              P(I) = AMIN1(Pet,PGrady)
              IF (Vr.LT.3.9*3281) THEN
                RC(I) = P(I)/2.*(Vr/3281./3.8)**(-1.32)
              ELSE
                RC(I) = P(I)/2.*(Vr/3281./4.6)**.2
              END IF
            ELSE
              Pet = D*2.33E-5*(Vr*COS(Theta))**1.16
              P(I)= AMIN1(Pet,PGrady)
              RC(I)=P(I)/(1-EXP(-Vr/5578.))
            END IF
C           *********************************************************

C           *** COMPUTE AVERAGE IMPACTS WITHIN CRATER**************
            AC(I)=PI*RC(I)**2/F1
            AC(I)=DMIN1(Ac(I),.999999999999D0*AS)
            PCR(I)=DMIN1(1D0,AC(I)/AS)
            LAMBDA(I)=NR(I)*AC(I)/AS
            SigSq(I) = Lambda(I)*(1D0-PCR(I))
            Sigma(I) = SQRT(SigSq(I))
C           *********************************************************
 2030     CONTINUE
C         *** New Version eight section*************************
C         *** This subroutine finds the minimum number of each***
C         *** size particle that must impact in one crater to ***
C         *** penetrate the plate ,how many craters they are in,*
C         *** and how many fragments are involved in shallower***
C         *** craters.******************************************
          CALL PenK(Plate,Thick,NBin,Maxk)
          CALL Countr
     :         (NBin,P,Thick(Plate),Maxk,Intact,Nr,Ac,As)
C         *** Number and Area of Holes and Residual Particles****
          HoArea=As*(1.D0-Intact)
          PlateM = HoArea*Thick(Plate)*Dense(Plate)*32.2
          AllMas=0.
          DO 2110 I=1,NBin-1
            AllMas=FrMass(I)*Nr(I)+AllMas
 2110     CONTINUE
          IF (AllMas.GT.0) THEN
            DO 2210 I=1,NBin-1
              NR(I)=NR(I)*(1.+PlateM/AllMas)
 2210       CONTINUE
```

```
          END IF
C         *** Separate calculation for LRM****
C         *** This option is for single penetrators*************************

          Call ResVel
     :      (Vr,ViLRM,RF(NBin),PDense(PrMat),Dense(TarMat(Plate)),
     :      Thick(Plate),Theta,PrMat,TMatSp(Plate))

C     ** Convert V to Km/S and MProj to grams
          ViX = Vi/3281.
          MProjX = LRM * 454.

C     ** Largest Residual Mass **
          ToverD=Thick(Plate)/2./Rf(NBin)
          CALL RFMax(MProjX,PDense(PrMat),ViX,RF(NBin),ToverD,MRMax,
     :                PSondV(PrMat),PrMat,Vc,Theta,FrTuff(PrMat))
          LRM=MrMax/454
          RF(NBin)=(LRM*3./(PDense(PrMat)*32.2*4.*PI))**(1./3.)

C         ****************************************************************
C         *** Test for Pennon and End of Run***********************
          IF (HoArea.LT..000000069.AND.Vr.LE.0.00001) THEN
C          IF (HoArea.LT.AC(1)) THEN
            Pennon = .FALSE.
            RETURN
          ELSE IF (Plate.EQ.10.OR.Thick(Plate+1).EQ.0.0) THEN
            Pennon = .TRUE.
            RETURN
          END IF
          ViLRM=Vr
          LastSp=SumSp
          SumSp=SumSp+Space(Plate)
          As=As*(SumSp/LastSp)**2
C         ***********************************************************
C         ************************END OF SHATTER EVALUATION**************
        END IF
 2040 CONTINUE
      RETURN
      END

      SUBROUTINE RFMax(M,RhoP,V,RP,ToverD,MrMax,C,PrMat,Vc,Theta,Kic)
C     M       Initial Impactor Mass                            Grams
C     MrMax   Mass of the Largest Residual Particle            Same as above
C     RhoP    Impactor Density                                 Slugs/Ft^3
C     Rp      Initial Impactor Radius (equivalent sphere)      Ft
C     V       Impactor Velocity                                Km/Sec
C     Shock   Velocity                                         Ft/Sec
C     Kic     Fracture Toughness                               MPa m^.5
C     Theta   Approach angle                                   Radians
C     PrMat   Projectile Material 1:Steel, 2:Aluminum ,3:Ice
      REAL Kic,k,MrMax,M,MrOMs,MrOMsC
      REAL MrMaxP,MrOMsP
      REAL MrMaxT,MrOMsT
      REAL RhoP,V,RP,ToverD,C,FTovrD,Vc
      INTEGER PrMat
      k=4.18E6
      IF (ToverD.LT..1) THEN
        FTovrD=1180.
      ELSE IF (ToverD.GE..1.AND.ToverD.LT..2) THEN
        FTovrD=697*ToverD**(-.23)
      ELSE IF (ToverD.GE.:2.AND.ToverD.LT..4) THEN
        FTovrD=244*ToverD**(-.881)
      ELSE IF (ToverD.GE..4.AND.ToverD.LT..8) THEN
        FTovrD=1500*ToverD**1.1
      ELSE
        FTovrD=1170.
      END IF
      IF (PrMat.EQ.1) THEN
        FTovrD=FTovrD*7.78
      END IF
      MrOMs = k*(Kic/(RhoP*32.2*C))**2/V**2/(RP*2*12*2.54)*FTovrD
      MrOMsT = MrOMs*COS(Theta)**(-2)
      MrOMsP = MrOMs*COS(Theta)
      Vc =(8.47E4*MrOMs*V**2)**(1/10.93)
      IF (V.LT.Vc) THEN
        MRMaxP=M*MrOMsP
      ELSE
        MrOMsC=1.18E-5*(Vc)**8.93
        MrMaxP=M*MrOMsC*(V/Vc)**(-5.5)
      END IF
      IF (V*COS(Theta).LT.Vc) THEN
        MRMaxT=M*MrOMsT
      ELSE
        MrOMsC=1.18E-5*(Vc)**8.93
        MrMaxT=M*MrOMsC*(V*COS(Theta)/Vc)**(-5.5)
      END IF
      MrMax = AMAX1(MrMaxP,MrMaxT)
      IF (PrMat.EQ.1) MrMax=MrMaxP
      MrMax = AMIN1(MrMax,M*.9999)
      RETURN
      END

      SUBROUTINE PenK(Plate,Thick,NBin,Maxk)
      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk(5),SumPr(5),Pk2
      DOUBLE PRECISION As,Ac,TestAc
      INTEGER Bin,Plate,MaxK(5),I,BinsDo,NBin
      REAL Thick(10),P,Vc
      LOGICAL BINDON(5)
      INTEGER k(5),MinI
      DOUBLE PRECISION kProbs(5,0:15)
      REAL LnPk,LnkFac
      COMMON /Count/kProbs
      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)
      COMMON /Crater/As,Ac(5),P(5)

C     *** k(NBin) is the number of each particle size that must ****
C     *** impact in one crater to make a hole.*******************
C     *** This loop finds the fraction of the plate that is not hit by ******
C     *** a fragment of size I (P(0)).  BinDon are initialized***
      TestAc=1D0-1D-3*Ac(1)/As
      BinsDo=0
      DO 10 I=1,NBin
        k(I)=0D0
        CALL Prs(I,Pk(I),k,NBin)
        kProbs(I,0)=Pk(I)
        SumPr(I)=Pk(I)
C       *** If there is not room for one more crater on the plate then ******
C       *** stop using this particle size **********************************
        IF (SumPr(I).GT.TestAc.OR.
     :         (K(I)*P(I)).GT.Thick(Plate)) THEN
          BinDon(I)=.TRUE.
          BinsDo=BinsDo+1
          kProbs(I,1)=1D0-kProbs(I,0)
        ELSE
          BinDon(I)=.FALSE.
        END IF
C       ********************************************************************
  10  CONTINUE
C     END DO
C     *********************************************************************
C     *** This subroutine finds which fragment size has the least SumPr *****
C     *** SumPr is the fraction of the plate that has craters of depth <= k**
C     *********************************************************************
C     *** This loop sums up the area of the plate that is not penetrated,****
C     *** while keeping the area covered by craters of depth <=k for each ***
C     *** size approximately equal.*********************************************
 20   IF (BinsDo.GE.NBin-1) GOTO 30
        CALL MiniI(MinI,BinDon,NBin,SumPr)
        k(MinI) = k(MinI)+1
C       *** This subroutine Calculates the fraction of the plate that is ****
C       *** covered by craters from exactly k particles of size MinI ********
        CALL PrS(MinI,Pk(MinI),k,NBin)
C       ********************************************************************
C       *** Add up the fraction of the spray area accounted for, and the ****
C       *** fraction of the particles used so far***************************
        SumPr(MinI)=SumPr(MinI)+Pk(MinI)
        kProbs(MinI,k(MinI))=Pk(MinI)
C       *****************************************************************
C       *** If there is not room for one more crater on the plate then ******
C       *** stop using this particle size***********************************
        IF (SumPr(MinI).GT.TestAc.OR.
     :         (K(MinI)*P(MinI)).GT.Thick(Plate)) THEN
          BinDon(MinI)=.TRUE.
          BinsDo=BinsDo+1
          Maxk(MinI)=K(MinI)
        END IF
C       ************************************************************************
      GOTO 20
 30   CONTINUE
       K(NBin) = 1
       kProbs(NBin,1)=1D0-kProbs(NBin,0)
       Maxk(NBin) = 1
C       ************************************************************************
      RETURN
      END

      SUBROUTINE MiniI(MinI,BinDon,NBin,SumPr)
      LOGICAL BinDon(5)
      INTEGER BinsDo
      DOUBLE PRECISION SumPr,MinSum
      DIMENSION SumPr(5)
      MinSum=1D0
      DO 10 I=1,NBin-1
        IF (.NOT.BinDon(I).AND.SumPr(I).LT.MinSum) THEN
          MinI=I
          MinSum=SumPr(I)
        END IF
   10 CONTINUE
C     END DO
      RETURN
      END

      SUBROUTINE PrS(I,Pk,k,NBin)
      DOUBLE PRECISION LastTe(5),PkMin,FPi
      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk,SumPr
      DOUBLE PRECISION As,Ac
      DIMENSION k(5)
      DIMENSION PkMin(5),kMin(5)
      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)
      FPi=1D0/SQRT(2D0*3.1415926D0)
      IF (Nr(I).GE.50.AND.Lambda I).LT.5) THEN
C       *** This section calculates the poisson approximation to ***
C       *** the binomial distribution*******************************
        IF (K(I).EQ.0)THEN
C         *** Initialize P(k) for first value***********************
          Pk=DEXP(-Lambda(I))
C         **********************************************************
        ELSE
C         *** Calculate P(k) from P(k-1)****************************
          Pk=LastTe(I)*Lambda(I)/K(I)
C         **********************************************************
        END IF
        LastTe(I)=Pk
C       ************************************************************
      ELSE IF(Pcr(I).GT..1.AND.Pcr(I).LT..9.AND.
     :        Nr(I).GT.9./(Pcr(I)*(1-Pcr(I))))THEN
C       *** This section calculates the normal approximation to ****
C       *** the binomial distribution*******************************
        Pk=FPi/Sigma(I)/DEXP(1/(2.D0*SigSq(I)*(k(I)-Lambda(I))**2))
C       ************************************************************
      ELSE
C       ***This section calculates the binomial distribution**********
        IF (k(I).EQ.0) THEN
C         *** Find least non-zero P(k) and corresponding k************
          CALL SetBin(I,PkMin,kMin,NBin)
          LastTe(I)=PkMin(I)
          IF (kMin(I).EQ.0) Pk=PkMin(I)
C         ************************************************************
        ELSE IF (k(I).GT.kMin(I)) THEN
C         *** calculate the non-zero P(k)s****************************
          Pk = LastTe(I)*Pcr(I)/(1D0-Pcr(I))*(Nr(I)-k(I)+1D0)/k(I)
          LastTe(I)=Pk
C         ************************************************************
        ELSE IF (k(I).EQ.KMin(I)) THEN
          Pk = PkMin(I)
        ELSE
          Pk = 0D0
        END IF
```

```
C           **********************************************************************
      END IF
      RETURN
      END
      SUBROUTINE SetBin(I,PkMin,kMin,NBin)
C     *** The Probability of the mean case occuring is computed ******
C     *** in subroutine Binomial.  P(kMin) is calculated from ********
C     *** this P(Lambda).  The magnitude of P(kMin) is arbitrarily ***
C     *** chosen to be about 1*10^-7.*********************************

      COMMON /Prob/Nr(5),Pcr(5),Lambda(5),SigSq(5),Sigma(5)

      DOUBLE PRECISION Nr,Pcr,Lambda,SigSq,Sigma,Pk,SumPr,N1,PkMin,P1
      DIMENSION PkMin(5),kMin(5)
      k=Lambda(I)
      N1=Nr(I)
      P1=Pcr(I)
      CALL Binomi(k,N1,Pk,P1)
 1    IF (.NOT.(Pk.GT.1D-13.AND.k.GE.1)) GOTO 10
        Pk = Pk*(1D0-Pcr(I))/Pcr(I)*k/(Nr(I)-k-1D0)
        k = k-1
      GOTO 1
 10   CONTINUE
C     END DO
      kMin(I)=k
      PkMin(I)=Pk
      RETURN
      END

      SUBROUTINE Binomi (k1,Nr,Pk,Pcr)
      DOUBLE PRECISION Nr
      DOUBLE PRECISION Pk,Pcr,Lambda,LastPk,Qcr
      DOUBLE PRECISION Top,Lower1,Lower2,k,Expon
      Lambda=k1
      k=k1
C     **********Binomial Distribution, Calculates P(k)************
      Lower1 = 0d0
      Lower2 = 0d0
      Top = Nr-k
      Pk = 1d0
      Qcr = 1D0-Pcr
      LastPk = 0D0
      IF (Qcr.EQ.1.D0) THEN
        IF (k.EQ.0.D0) THEN
          Pk=1.D0
        ELSE
          Pk=0.D0
        ENDIF
      ELSE
 3040   IF (Lower1+Lower2+Top.GE.(2*Nr).OR.Pk.EQ.LastPk) GOTO 3030
          LastPk = Pk
 3060     IF (Top.GE.Nr.OR.Pk.GE.1E23) GOTO 3050
            Top = Top + 1D0
            Pk = Pk * Top
          GOTO 3060
 3050     CONTINUE
 3080     IF (Lower1.GE.k.OR.Pk.LE.1E-20) GOTO 3070
            Lower1 = Lower1 + 1d0
            Pk = Pk /Lower1*Pcr
          GOTO 3080
 3070     CONTINUE
          IF (Lower1.GE.k.AND.Lower2.LT.Nr-k.AND.Pk.GT.1D-27) THEN
            Expon=DMIN1(((-28.D0-DLOG10(Pk))/DLOG10(Qcr)),(Nr-k-Lower2))
            IF (Expon.GT.0) Pk=Pk*Qcr**Expon
            Lower2 = Lower2+Expon
          ENDIF
          GOTO 3040
 3030   CONTINUE
      END IF
C     ************************************************************
      RETURN
      END
```

```

      SUBROUTINE MasChr (V,M,ToverD,Theta,RP,RhoP,RhoT,C,NBin,
     :                   MenRad,Nr,MrMax,MAvg,PrMat,FrTuff)
      DOUBLE PRECISION NR(5)
      REAL MProj,M,MRMax,MAvg,MenRad(5),MPlate
      REAL MasLim,NrmF,NrmS,BinMen(5),BinMas(0:5)
      INTEGER Bin,PrMat
      PI = 3.14159

C     This subroutine divides the residual mass into bins of equal mass
C     The number of fragments in each bin are also noted

C     ** Convert V to Km/S and MProj to grams
      Vi = V/3281.
      MProj = M * 454.
      MPlate = M * 454.

C     ** Largest Residual Mass **

      IF (PrMat.EQ.3) THEN
        CT = 17569
        RhoT = 5.39
        CALL RFMax(MPlate,RhoT,Vi,RP,ToverD,MRMax,
     :           CT,PrMat,Vc,Theta,FrTuff)
      ELSE
        CALL RFMax(MProj,RhoP,Vi,RP,ToverD,MRMax,
     :           C,PrMat,Vc,Theta,FrTuff)
      END IF

C     ** Average Residual Mass **
      CALL AvgRes (ToverD,Theta,MPlate,Vi,Alfa,MAvg)
      IF (Vi.GT.Vc) MAvg=MAvg*(Vi/Vc)**(-5.5)

C     ** Parameters and Normalization Constants for Weibull Distribution **
      CALL ShCons (ToverD,Theta,MProj,Vi,MAvg,
     :                         bS,sS,bF,sF,NrmS,NrmF,MRMax)

C     ** Size Shatter Begins **
      CALL Shhold (MProj,Vi,TD,Theta,MAvg,bF,sF,bS,sS,
     :                     NrmF,NrmS,FrgLim,MasLim,MRMax)

C     ** Upper Bound and Mean Fragment weight for each Bin **
      CALL BinLim(NBin,MAvg,MRMax,BinMas,BinMen,MProj,
     :                  sF,bF,sS,bS,NrmS,NrmF,MasLim)

C     ** Loop Determining Number of Fragments in each Bin in Shatter Regime *
      DO 10 Bin = 1,NBin-1
        Nr(Bin)=(BinMas(Bin)-BinMas(Bin-1))/BinMen(Bin)
 10   CONTINUE
C     END DO
      Nr(NBin) = 1
C     ** Convert Bin Masses into lbs from grams then to Radius in Feet**
      DO 20 Bin = 1,NBin
        BinMen(Bin)=BinMen(Bin)/454.
        MenRad(Bin) = (BinMen(Bin)*3./(RhoP*32.2*4.*PI))**(1./3.)
  20  CONTINUE
C     END DO
      MrMax=MrMax/454.
      MPlate = MPlate/454.
      MProj = MProj/454.
      RETURN
      END
```

C-4

```

        SUBROUTINE MasDis (M,MAvg,b,s,NrmC,MasSum)
C
        THIS PROGRAM COMPUTES SHATTER Mass distributions
        STEEL CUBES ON ALUMINUM PLATE DATA.
             V = IMPACT VELOCITY     (FEET/SECOND)
           TD = T OVER D
             M = PROJECTILE MASS     (GRAINS)
         THETA = ANGLE OF OBLIQUITY
          ALPHA=ANGLE OF IMPACT
C
        REAL MAvg,M,NrmC,MasSum
        W = M/MAvg
C       **Mass in Bin**
      determ=b+s*LOG(W)
      IF (Determ.LT.-80) THEN
        MasSum=0
      ELSE IF (Determ.LT.4) THEN
        MasSum = NrmC*(1-EXP(-EXP(Determ)))
      ELSE
        MasSum=NrmC
      END IF
      RETURN
      END
```

```
C           SSSSSSSSSSSSSS  SALVO's    CHANGES  SSSSSSSSSSSS

          SUBROUTINE ShCons(TD,Theta,M,Vi,MAvg,
         :                                 bS,sS,bF,sF,NrmCS,NrmCF,LRM)
          REAL M,NrmCS,NrmCF,MT2,MAvg,IntegS,IntegF,LRM
              sS=1.8-.04*TD/COS(THETA)-.042*M
         &        +.34*(COS(2.*ALPHA)**2.)-(1-EXP(-.64*Vi))
              bS=-2.3-1.1*TD/COS(THETA)+.0675*M
         &        -.27*Vi+1.4*(COS(2.*ALPHA)**2.)
              sF=1.38-.510*TD/COS(THETA)+.036*M
         &        +3.31*(COS(2.*ALPHA)**2.)-(1-EXP(-.390*Vi))
              bF=-1.17+.313*TD/COS(THETA)+.0675*M
         &        +.508*Vi-1.41*(COS(2.*ALPHA)**2.)

C         ** Total mass in shatter regime **
         MT2 =(-.957+EXP(-.0013*M))*(Vi**(.38*M+2.5))
         MT2 = AMAX1(MT2,1E-20)
         MT2 = AMIN1(MT2,M)
C           **Normalization to Largest Residual Mass**
C            IntegS = 1-EXP(-EXP(bS)*(M/MAvg)**sS)
C            IntegF = 1-EXP(-EXP(bF)*(M/MAvg)**sF)
C            NrmCS = MT2/IntegS
C            NrmCF = M  /IntegF
         detrml=bS+sS*ALOG(LRM/MAvg)
         IF (Detrml.LT.-15) THEN
           NrmCS=1e-30
         ELSE IF (Detrml.LT.4) THEN
           IntegS = 1-EXP(-EXP(Detrml))
           NrmCS = MT2/IntegS
         ELSE
           NrmCS = MT2
         END IF
         detrm2=bF+sF*LOG(LRM/MAvg)
         IF (Detrm2.LT.-15) THEN
           NrmCS=1e-30
         ELSE IF (Detrm2.LT.4) THEN
           IntegF = 1-EXP(-EXP(Detrm2))
           NrmCF = M/IntegF
         ELSE
           NrmCF = M
         END IF
           if(sF)1,2,2
1          sF=0.
2          if(sS)3,4,4
3          sS=0.
4        RETURN
         END
```

```

      SUBROUTINE Shhold (MProj,Vi,TD,Theta,MAvg,bF,sF,bS,sS,
     :                   NrmF,NrmS,FrgLim,MasLim,LRM)
      REAL LowLim,Mean,Mfs,Mss,MAvg,MasLim,MProj,NrmF,NrmS,M,LRM
      HiLim = MProj
      LowLim = 0
      DO 10 I=1,20
        Mean = (HiLim+LowLim)/2
        CALL MasDis (Mean,MAvg,bF,sF,NrmF,Mfs)
        CALL MasDis (Mean,MAvg,bS,sS,NrmS,Mss)
        IF (Mfs.GT.Mss) THEN
          HiLim = Mean
        ELSE
          LowLim = Mean
        END IF
   10 CONTINUE
C     END DO
      FrgLim = Mean
      MasLim = AMIN1(Mss,MProj-LRM)
      RETURN
      END
      SUBROUTINE BinLim (NBin,MAvg,LRM,BinMas,BinMen,MProj,
     :                      sF,bF,sS,bS,NrmS,NrmF,MasLim)
      REAL MAvg,LRM,MProj,MasLim,NrmS,NrmF,BinInt,MasMen,MFrac(0:4)
      DIMENSION BinMen(5),BinMas(0:5),MasMen(5)
      INTEGER Bin,ShtBin
      DATA MFrac/0.,.25,.5,.75,1./
      ShtBin=0
      DO 5 Bin=1,NBin-1
        BinMas(Bin) = (MProj-LRM)*MFrac(Bin)
        MasMen(Bin) = (MProj-LRM)*(MFrac(Bin)+MFrac(Bin-1))/2.
        IF (BinMas(Bin).LE.MasLim) ShtBin=Bin
    5 CONTINUE
C     END DO
C     ** Bin Limit and Mean for Shatter **
      DO 10 Bin = 1,ShtBin
       BinMen(Bin) = MAvg*(-EXP(-bS)*DLOG(1D0-MasMen(Bin)/NrmS))**(1/sS)
   10 CONTINUE
C     END DO
C     ** Bin Limit and Mean for First Bin in Fracture **
      Bin = ShtBin+1
      IF (Bin.LE.NBin-1) THEN
        IF (MasLim.GE.MasMen(Bin)) THEN
C         ** Bin Mean in Case it is Less Than the Fracture Threshold **
         BinMen(Bin)=MAvg*(-EXP(-bS)*DLOG(1D0-MasMen(Bin)/NrmS))**(1/sS)
        ELSE
         BinMen(Bin)=MAvg*(-EXP(-bF)*DLOG(1D0-MasMen(Bin)/NrmF))**(1/sF)
        END IF
      END IF
C     ** Bin Limit and Mean for Fracture **
      DO 20 Bin = ShtBin+2,NBin-1
       BinMen(Bin) = MAvg*(-EXP(-bF)*DLOG(1D0-MasMen(Bin)/NrmF))**(1/sF)
   20 CONTINUE
C     END DO
C     ** Bin Mean for largest bin is LRM (Largest Residual Mass) **
      BinMen(NBin) = LRM
      RETURN
      END
```

```
      SUBROUTINE AvgRes (TоverD,Theta,M,V,Alfa,MAvg)
      REAL MAvg,M
C     ** Average Fragment Mass **
      MAvg = .0109-.00879*ToverD/COS(Theta)+.000506*M-.00428*V+
     :        .0110*COS(2*Alfa)**2
      MAvg = AMax1(MAvg,0.005)
      RETURN
      END
```

```fortran

      SUBROUTINE Countr(NBin,P,Thick,Maxk,Intact,Nr,Ac,As)
      INTEGER Bin,NBin,SmBin
      INTEGER Digit(5),HoMin(5),I,Maxk(5)
      REAL Thick,HoDept,P(5)
      DOUBLE PRECISION Probs(5,0:15)
      DOUBLE PRECISION Intact,ITArea,ITArS1
      DOUBLE PRECISION Nr(5),Ac(5),As
      DOUBLE PRECISION ArHol1,ArHol2,ArHol3
      DOUBLE PRECISION ArHol4,ArHol5
      DOUBLE PRECISION PrTemp,DigTp1,DigTp2,DigTp3
      DOUBLE PRECISION          DigTp4,DigTp5
      LOGICAL UnInc,OBin1D
      COMMON /Count/Probs
      DO 10 Bin = 1,NBin
        HoMin(Bin) = JMIN0(INT(Thick/P(Bin)+1),Maxk(Bin))
        Digit(Bin)=0
   10 CONTINUE
C     END DO
      ArHol1=0.D0
      ArHol2=0.D0
      ArHol3=0.D0
      ArHol4=0.D0
      ArHol5=0.D0
      Intact=0.D0
      Bin = 0
      HoDept = 0
      OBin1d = .FALSE.
C     DO WHILE (Bin.LE.NBin)
    2 IF (.NOT.(Bin.LE.NBin)) GOTO 20
        IF (HoDept.GT.Thick) THEN
          OBin1d = .FALSE.
          SmBin=1
C         DO WHILE (Digit(SmBin).EQ.0)
   21     IF (.NOT.(Digit(SmBin).EQ.0)) GOTO 210
            SmBin=SmBin+1
          GOTO 21
  210     CONTINUE
C         END DO
          Digit(SmBin)=0
          HoDept=0.
          Bin = SmBin+1
          UnInc = .TRUE.
C         DO WHILE (Bin.LE.NBin.AND.UnInc)
   22     IF (.NOT.(Bin.LE.NBin.AND.UnInc)) GOTO 220
            IF (Digit(Bin).LT.HoMin(Bin)) THEN
              Digit(Bin) = Digit(Bin)+1
              UnInc = .FALSE.
            ELSE
              Digit(Bin)=0
              Bin = Bin + 1
            END IF
          GOTO 22
  220     CONTINUE
C         END DO
          DO 230 I=SmBin+1,NBin
            HoDept=HoDept+Digit(I)*P(I)
  230     CONTINUE
C         END DO
        ELSE
          IF (OBin1d) THEN
            DigTp1=DigTp1+1
            ITArea = Probs(1,DigTp1)*ITArS1

            ArHol1 = ArHol1+ITArea*DigTp1
            ArHol2 = ArHol2+ITArea*DigTp2
            ArHol3 = ArHol3+ITArea*DigTp3
            ArHol4 = ArHol4+ITArea*DigTp4
            ArHol5 = ArHol5+ITArea*DigTp5

          ELSE
            ITArea=1.D0

            DigTp2=Digit(2)
            PrTemp = Probs(2,DigTp2)
            ITArea = PrTemp*ITArea

            DigTp3=Digit(3)
            PrTemp = Probs(3,DigTp3)
            ITArea = PrTemp *ITArea

            DigTp4=Digit(4)
            PrTemp = Probs(4,DigTp4)
            ITArea = PrTemp *ITArea

            DigTp5=Digit(5)
            PrTemp = Probs(5,DigTp5)
            ITArea = PrTemp *ITArea

            ITArS1=ITArea

            DigTp1=Digit(1)
            PrTemp = Probs(1,DigTp1)
            ITArea = PrTemp *ITArea

            ArHol1 = ArHol1+ITArea*DigTp1
            ArHol2 = ArHol2+ITArea*DigTp2
            ArHol3 = ArHol3+ITArea*DigTp3
            ArHol4 = ArHol4+ITArea*DigTp4
            ArHol5 = ArHol5+ITArea*DigTp5

          END IF
          Intact=Intact+ITArea
          Bin = 1
          OBinld = .TRUE.
          UnInc = .TRUE.
C         DO WHILE (Bin.LE.NBin.AND.UnInc)
 24       IF (.NOT.(Bin.LE.NBin.AND.UnInc)) GOTO 240
            IF (Digit(Bin).LT.HoMin(Bin)) THEN
              Digit(Bin) = Digit(Bin)+1
              HoDept = HoDept+P(Bin)
              UnInc = .FALSE.
            ELSE
              OBinld = .FALSE.
              HoDept = HoDept-Digit(Bin)*P(Bin)
              Digit(Bin)=0
              Bin = Bin + 1
            END IF
          GOTO 24
 240      CONTINUE
C         END DO
        END IF
      GOTO 2
 20   CONTINUE
C     END DO
        Nr(1)=DMAX1(0D0,Nr(1)-ArHol1*As/Ac(1))
        Nr(2)=DMAX1(0D0,Nr(2)-ArHol2*As/Ac(2))
        Nr(3)=DMAX1(0D0,Nr(3)-ArHol3*As/Ac(3))
        Nr(4)=DMAX1(0D0,Nr(4)-ArHol4*As/Ac(4))
        Nr(5)=DMAX1(0D0,Nr(5)-ArHol5*As/Ac(5))
      RETURN
      END

      SUBROUTINE ResVel
     :  (Vr,V0,RP,RhoP,RhoT,Thick,Theta,PrMat,TarMat)
C       This is the JTCG Residual Velocity Formula
      INTEGER PrMat,TarMat
      PresAr = 3.14159*RP**2
      V0cm = V0/.03281
      Weight = 4./3.*3.14159*RP**3*RhoP
      V50 = BalLim(RP,RhoP,RhoT,Thick,Theta,PrMat,TarMat)
      Q4 = RhoT*Thick*PresAr/(Weight*COS(Theta))
      VrSq = AMAX1(0.,V0cm**2-V50**2)
      Vr = SQRT(VrSq) / (1.+Q4)
      Vr = Vr*.03281
      RETURN
      END
```

```

      FUNCTION BalLim
     :                     (RP,RhoP,RhoT,Thick,Theta,PrMat,TarMat)
      INTEGER PrMat,TarMat
      IF (PrMat.EQ.3) THEN
        BalLim = 2.45*3281*(Thick/12./2.54)**(-.15)*
     :      (Thick/RP/2.)*(RhoT/RhoP)**.64*(1./COS(Theta))**1.01
      ELSE
C       This is the JTCG V50 formula
        PresAr = 3.14159*RP**2
        Weight = 4./3.*3.14159*RP**3*RhoP*32.2
        W0 = .0143
        IF (PrMat.EQ.1) THEN
          IF (TarMat.EQ.2) THEN
            Cbf = 41300.
            Bf  =  .941
            H   = 1.098
            F   = -.038
          ELSE
            Cbf = 80600.
            Bf  =  .963
            H   = 1.286
            F   = -.057
          END IF
        ELSE
          Cbf = 92800
          Bf  =  .972
          H   = 1.01
          F   = -.105
        END IF
        Q8 = RhoP*32.2*Thick*PresAr/Weight
        Q11 = RhoP*32.2*Thick*PresAr/W0
        BalLim = Cbf*Q8**Bf/COS(Theta)**H*Q11**F
      END IF
      RETURN
      END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE CRITDIA
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Critdia determines the diameter of the sphere that just penetrates
C  the given wall configuration at the given impact angle and velocity.
C  The subroutine performs a linear interpolation using the appropriate
C  values from the Response array to estimate the critical diameter.
C
C  note: for variables contained in the common block refer to the main
C        listing for definition
C
C
C  Variable list
C
C    beta = impact angle , measured from the normal , deg
C    b1 = the impact angle nearest to the actual impact angle
C         in the Response array , but less than the actual , deg
C    ib1 = location of b1 in the Response array
C    ib2 =    "      " b2  "  "      "        "
C    iv1 =    "      " v1  "  "      "        "
C    iv2 =    "      " v2  "  "      "        "
C    r1 =  intermediate variable
C    r2 =        "          "
C    r11 = value in Response array at location iv1,ib1,pid
C    r12 =  "     "     "       "   "   "      iv1,ib2,pid
C    r21 =  "     "     "       "   "   "      iv2,ib1,pid
C    r22 =  "     "     "       "   "   "      iv2,ib2,pid
C    v1 = impact velocity nearest the actual impact velocity in the
C         Response array , but still less than the actual
C
C
C
C
      INTEGER*2 IB1,IB2,IV1,IV2,PID
C
      INCLUDE 'COMMON3.BLK'
C
      PARAMETER (PI=3.1415926536)
C
C  Determine the location of the nearest velocity to the actual velocity
C  in the Response array, but still less then the actual
C
      IV1=VR/VINC
C
C  Check that the location is inside the array
C
      IF ( IV1.LT.1 .OR. IV1.GT.NV ) THEN
         WRITE ( 6,10 )VR
  10     FORMAT ( /1X,'THE RELATIVE VELOCITY (VR) IS OUTSIDE OF THE',
     1            ' RESPONSE ARRAY BOUNDS VR (KM/SEC) = ',E12.5)
         STOP
      END IF
C
C  Set the location of the velocity just greater than the actual velocity
C
      IV2=IV1+1
C
```

```
C Check that the location is inside the array
C
    IF ( IV2.GT.NV ) IV2=IV1-1
C
C Calculate the value of the velocity at location iv1
C
    V1=IV1*VINC
C
C Determine the impact angle in deg
C
    BETA=ACOS(CBETA)*180.0/PI
C
C Determine the location of the nearest impact angle to the actual
C impact angle in the Response array, but still less than the actual
C
    IB1=BETA/BINC+1
C
C Check that the location is inside the array
C
    IF ( IB1.LT.1 .OR. IB1.GT.NB ) THEN
      WRITE ( 6,20 )BETA
  20  FORMAT ( /1X,'IMPACT ANGLE (BETA) IS OUTSIDE THE BOUNDS OF',
   1           ' THE RESPONSE ARRAY BETA (DEG) = ',E12.5)
      STOP
    END IF
C
C Set the location of the impact angle in the Response array that is
C just greater than the actual
C
    IB2=IB1+1
C
C Check that the location is inside the array
C
    IF ( IB2.GT.NB ) IB2=IB1-1
C
C Calculate the value of the impact angle at location ib1 in the Response
C array
C
    B1=(IB1-1)*BINC
C
C Determine the property id
C
    PID=ID(2,NEL)
C
C Get the four values that surround the actual value in the Response
C array
C
    R11=RTABLE(IV1,IB1,PID)
    R12=RTABLE(IV1,IB2,PID)
    R21=RTABLE(IV2,IB1,PID)
    R22=RTABLE(IV2,IB2,PID)
C
C Using linear interpolation, estimate the critical diameter
C
    R1=(R12-R11)*((BETA-B1)/BINC)+R11
    R2=(R22-R21)*((BETA-B1)/BINC)+R21
C
    DIAM=(R2-R1)*((VR-V1)/VINC)+R1
C
C Finished , return
C
    RETURN
C
```

```
0149        END
```

```
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      SUBROUTINE FLUX
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C  Flux calculates the meteoroid or debris flux for the given critical
C  diameter based on analysis type.
C
C
C  note: for variables contained in the common block referr to the main
C        listing for definition
C
C  Variable List
C
C   ddiam = diam in double precision , cm
C   ge = gravity focusing factor
C   intercept = intercept of the flux equation
C   mass = critical meteoroid mass, g
C   mden = meteoroid density, g/cc
C   re = earth's radius (including 100km atmosphere), km
C   slope = slope of the flux equation
C
C
      INCLUDE 'COMMON3.BLK'
C
      REAL*8 DDIAM,GE,INTERCEPT,LD,MASS,MDEN,PI,RE,SLOPE
C
      PARAMETER (PI=3.141592653589793238D0)
C
C  Set  mden
C
C
      MDEN=0.50D0
C
C  Calculate the focusinng factor, equation
C  is from JSC-30000
C
      RE=6478.0D0
      GE=0.568D0+0.432D0*(RE/(RE+ALT))
C
C  Convert diam to double precision
C
      DDIAM=DIAM
C
C  Calculate the flux
C
      IF ( ITYPE.EQ.1 ) THEN
C
C  For debris use JSC-20001, use stated equations for diameters
C  less then 1 cm , for those greater use third order fit of the
C  curve for region up to 5 cm .
C
C  The log of the flux varies linearly between 400 and 500 km according
C  to D Kesseler of JSC.
C
         LD=DLOG10(DDIAM)
         IF ( DIAM.LT.5.0 ) THEN
            IF ( DIAM.LT.1.0 ) THEN
               SLOPE=-0.0010D0*ALT-2.0200D0
            ELSE
               SLOPE=-0.0022D0*ALT-0.1400D0
            END IF
            INTERCEPT=+0.0036D0*ALT-7.26D0
            FLX=10.0D0**(SLOPE*LD+INTERCEPT)
         ELSE
            WRITE ( 6,100  )
  100       FORMAT ( /1X,'DIAMETER IS GREATER THAN 5 CM LIMIT')
            STOP
         END IF
C
C  Correct Flux for differance in Boeing and Nasa definetion
C
         FLX=FLX*4.0D0
C
      ELSE
C
C  For meteoroids use JSC-3000,  E-06g < mass < 1g
C
         MASS=PI*(DDIAM**3)/6.0D0*MDEN
         FLX=10.0D0**(-14.37D0-1.213D0*DLOG10(MASS))
C
C  Account for earth shielding and gravity focusing , also convert to
C  number of impacts per sq-m per year
C
         FLX=FLX*GE*3.15576D07
C
      END IF
C
      RETURN
C
      END
```

COMMON3.BLK

```
C
C   Common Block for Contour Ver 2.0 6/2/87
C
C   ielm - max number of elements
C   ith - max number of threats
C
      PARAMETER (IELM=9000,ITH=400)
C
      INTEGER*2 IT,ITYPE,NB,NC,NEL,NELM,NT,NV,EXPOSED(ITH),
     1          POINT(IELM,ITH)
C
      INTEGER*4 NR,ID(2,IELM),RANGE(2)
C
      REAL*4 BINC,CBETA,DIAM,VINC,AREA(IELM),GEOMETRY(IELM,ITH),
     1       RTABLE(70,90,10),THREAT(4,ITH)
C
      REAL*8 ALT,ETIME,FLX
C
      COMMON ALT,BINC,CBETA,DIAM,ETIME,FLX,IT,ITYPE,NB,NC,NEL,NELM,NR,
     1       NT,NV,VR,VINC,AREA,EXPOSED,GEOMETRY,ID,POINT,RANGE,RTABLE,
     2       THREAT
C
```